Naval Research Laboratory

Washington, DC 20375-5320

NRL/MR/7130--95-7649

GENAH User's Guide and Reference Manual Version 1.0

KARL B. WASHBURN

Physical Acoustics Branch
Acoustics Division

January 17, 1995

Approved for public release; distribution unlimited.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave Blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	January 17, 1995	Memorandum Report 7/1/94-9/26/94

4. TITLE AND SUBTITLE

GENAH User's Guide and Reference Manual Version 1.0

5. FUNDING NUMBERS

PE -62323N
PR -71-M107-X4

6. AUTHOR(S)

Karl B. Washburn

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Research Laboratory
Washington, DC 20375-5320

8. PERFORMING ORGANIZATION REPORT NUMBER

NRL/MR/7130--95-7649

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

CDNSWC
Bethesda, MD 20084

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

Generalized Nearfield Acoustical Holography, Algorithm, Computer Software

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

This document is the User's Guide and Reference Manual for GENAH, the Generalized Nearfield Acoustical Holography processing program. The User's Guide provides instructions for operating the GENAH programs. This includes installing and running the code, choosing input parameters and a discussion of the input and output file structure. The Reference Manual contains material explaining the theory of cylindrical nearfield holography and how this theory is implemented in the GENAH code. Sufficient detail is provided to allow a user with an operating knowledge of FORTRAN-77 to diagnose and alter the code to suit local needs. Appendices are included detailing the file formats, the file header, a list of operator inputs to GENAH, a list of all routines needed to compile and run the GENAH code, and a discussion of wavenumber filtering in holography.

14. SUBJECT TERMS	15. NUMBER OF PAGES
	124
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std 239-18
298-102

CONTENTS

List of Tables

List of Figures

Part I

User's Guide

GENAH USER'S GUIDE AND REFERENCE MANUAL VERSION 1.0

Chapter 1

Introduction

GENAH is the GEneralized Nearfield Acoustical Holography processing system. It is used to project acoustical pressure holograms taken in cylindrical coordinates to reconstruction surfaces both inside and outside the original hologram surface. It provides reconstructions in both the spatial and wavenumber domains, and its outputs include pressure, normal (radial) velocity and intensity.

First developed in 1985, GENAH is a direct outgrowth of planar nearfield holographic algorithms whose code dates back to 1980. The code has undergone many transitions in its lifetime. It was ported from its original PDP-11 progenitor under RT-11 to a VAX 11-785 running VMS where it underwent extensive modification to (approximately) the form seen today. In 1990, it was again ported, this time to a Silicon Graphics platform running Unix. Because of major differences in both the native number representations and the `FORTRAN` compilers, this latter port engendered a number of major changes to the subroutines that support the core GENAH algorithm. This allowed the code, running on an SGI machine, to read files in DEC number format if necessary for backward compatibility. The version included in this 1994 public release has undergone extensive (although mostly cosmetic) revision to provide a uniform user interface and to make the code more readable. Nonetheless, the interested user will recognize aspects of coding that harken back to the original (very limited) code used on the PDP-11.

The User's Guide is intended to provide the user/operator with the information necessary to install and intelligently operate the software. It opens with a short guide to installing the software on any Unix system with a

Manuscript approved December 9, 1994.

FORTRAN-77 compiler. A very brief tour of GENAH is then provided in the form of a sample run. Next, a thorough discussion of the primary processing options is undertaken. The user must understand which "direction" the processing is to proceed before intelligent parameter choices can be made. A more detailed description of the user inputs follows, with information on choosing the many parameters necessary to guide GENAH 's processing. Finally, the input and output files used by GENAH are described. This description is meant to allow the user to proceed with operating the software: very detailed descriptions of the files, their format and header specifications are provided as appendices.

The Reference Manual is intended to provide the user with an in-depth understanding of GENAH . The theory of Nearfield Acoustical Holography is introduced with references to the literature included. The specialization to cylindrical coordinates is demonstrated, including the Green's Function propagator and differences in interpretation of the wavenumber domain from planar systems. The implementation of the propagation algorithm is then discussed. Finally, a very thorough description is presented of the actual code included in this public release. This latter should be sufficient to allow for local modification of the code as needed to alter I/O, processing options, and so on.

Chapter 2

Installing GENAH

2.1 Routines in this Release

The files in this release consist of a set of `FORTRAN-77` routines, a Makefile for compiling and linking them, a sample variable filter file, a sample input hologram and a sample script file. Table 2.1 lists the files as they appear in the distribution (that is, in alphabetical order). Table 2.2 breaks these same files down into categories based on their purpose. Those files ending with a "`.f`" subscript are `FORTRAN` routines.

To install GENAH , simply make a suitable subdirectory in your filesystem and place the files there. If they were saved as one file from an E-mail file, you will need to use an editor to separate them. If they were loaded as a "`tar`" (backup tape) file, simply use the "`tar -xvf`" command to extract them. Once a directory listing shows the same files as those listed in Table 2.1, they are ready to be compiled and tested.

GENAH depends on a set of Bessel function routines to compute the propagators. The routines currently used at NRL are those provided by the VAXMATH collection of `FORTRAN` scientific subroutines, themselves descendants of the venerable SLATEC routines. All of the Bessel function routines and their support routines are included in this release. They should be compiled into an archive (library), referred to in the Makefile as "`bessel.a`". (If your library will be named differently, e.g. "`BESSEL.OLB`", then the make file should be altered accordingly.) If your installation includes the double–precision VAXMATH routines (compiled G–floating on VAXes), then they

Bytes	*File Name*
1231	`Makefile`
600	`Sample.Script`
1.4M	`Test.hol`
46	`Test.window`
905	`cang.f`
1077	`cvt_2byte_to_int4.f`
1187	`cvt_4byte_to_real4.f`
1081	`cvt_decf_ieeef.f`
1131	`cvt_int4_to_2byte.f`
1345	`cvt_real4_to_4byte.f`
2474	`fft.f`
5619	`filread_ieee.f`
3571	`filwrite_ieee.f`
40443	`genah.f`
983	`hanning.f`
2446	`header_ieee.f`
4697	`read_from_ihead.f`
1443	`tukey.f`
1159	`windo.f`
4677	`write_to_ihead.f`
2467	`wtrev.f`

Table 2.1: List of files in this release of GENAH , and their sizes in bytes.

may be substituted for the Bessel routines provided here. The Makefile (or compile command file) should be edited to reflect this before attempting to compile and link the files.

Important Note: The files `I1MACH.f` and `D1MACH.f` provided in this release are currently set to execute on `IEEE.MOST-SIG-BYTE-FIRST`-style CPUs, such as the SGI MIPS or Sun SPARC. If you will be using some other platform, you *must* edit these two files to reflect the CPU type on which the routines will be compiled and executed.

Purpose	File Name
Main Processing Routine	genah.f
Header Include File Data File I/O	header_ieee.f filread_ieee.f filwrite_ieee.f
Data File Header	read_from_ihead.f write_to_ihead.f
FP Number Conversion	cvt_2byte_to_int4.f cvt_4byte_to_real4.f cvt_int4_to_2byte.f cvt_real4_to_4byte.f cvt_decf_ieeef.f
Spatial Windowing	hanning.f tukey.f
Wavenumber Filtering	windo.f
Fourier Transforms	wtrev.f fft.f
Phase of Complex Number	cang.f

Table 2.2: GENAH files, grouped by purpose.

2.2 Compiling GENAH : the Makefile

GENAH is composed of `FORTRAN` routines. It will only execute if compiled with a `FORTRAN-77` or compatible compiler. It does not use any system–specific calls and is self-contained for execution on most systems.

The compilation of GENAH has been greatly simplified through the introduction of a "Makefile." If the system on which GENAH will run has the "Make" utility, simply enter `make genah` at the command line in your designated GENAH subdirectory. The routines will then be compiled with the native `FORTRAN-77` compiler and linked (or "loaded") into an executable image called "`genah`". On a VAX system, the "`FC`" variable in the Makefile will have to be changed to "`FORT`", and the "`OPT`" variable will have to be cleared (no options are needed by the VAX compiler).

If your system does *not* have the "Make" utility, you will have to compile and link the routines by hand (or with a script file). If you are using a

VAX system, no flags are needed. Simply use the `FORT` command on each file, then a `LINK` command with all the object files listed (with "`GENAH.OBJ`" first) separated by commas. This process can be simplified with a "`.COM`" file, which the user can base on the included Makefile and build using an editor.

If you are using a Unix system, use the `f77` (or its equivalent) command with a flag to generate object files (often "`-c`") and a flag to force static memory allocation (often "`-static`"). This latter flag is not strictly necessary, but it has been found to be the safest way to compile the GENAH code. The object files are then linked (or loaded) with the `f77` command again, using the "`-o`" option to generate the output name "`genah`". Alternatively, some systems use the `ld` command to link object files into an executable image. Use whichever is appropriate for your system. Again, see the Makefile for syntax and filenames; a shell script could be developed from it.

Note that GENAH has never been compiled above so-called "Level 1" optimization (where loop variables are scrutinized and some are placed in registers). We cannot guarantee the safe and accurate operation of GENAH for higher levels of optimization. There is no inherent reason why GENAH should not perform well under such optimization, but doing so would be the responsibility of the user.

2.3 Testing the GENAH Installation

Once the code has been extracted and compiled, it can be tested with the example files included in the distribution. In particular, an example Unix script file (`Sample.script`) is included to run the GENAH code, as well as a small hologram file (`Test.hol`) on which it operates. The user can simply execute the script from the command line or use it as a guide to run GENAH from within a screen shell to get a feel for using it.

Unfortunately, since hologram files are encoded in a particular binary format (see Appendix A), it is not a trivial matter to examine, list, plot or visualize them. The user *can* examine the ASCII output generated for each bin in processing `Test.hol` as a first-order check on the GENAH installation. Several plotting and visualization routines have been developed at NRL; they are not included in this software release because they are specific to the SGI 4D family of graphics workstations. Two options are available. First, the user could check with the authors to see if any of the currently available tools might be applicable to the user's system. Second, the user could (and probably should) locally develop plotting and listing software, using the I/O routines provided in this release (see Appendix D) to load holograms and plot or list their results.

Chapter 3

A Quick Tour of GENAH

The following is a very brief introduction to the GENAH software. In Section 3.1, the structure of the GENAH code is outlined in a manner that will familiarize the user with some of the choices to be made for the input parameters. In Section 3.2, a sample session with GENAH is listed and described. This session uses the same inputs as that contained in the script `Sample.script`, provided in this release.

3.1 Basic GENAH Structure

For detailed information on how the FORTRAN code in GENAH is laid out, refer to Chapter 9. For information on how that code implements the GENAH algorithm, refer to Chapter 8. For an in–depth discussion of holography theory and how it leads to an algorithm, see Chapter 7. All of these Chapters are in the Reference Manual.

GENAH is composed of two primary sections, a preamble and the main frequency loop. Within the preamble, the user opens the input hologram pressure file and defines all of the flags and parameters needed to execute GENAH on the input data. The main loop steps through the holograms and processes each one according to the directives defined in the preamble. The resulting reconstructions are then written to the appropriate output files.

Opening the input data file sets most of the parameters needed for hologram processing. These include the size of the hologram mesh itself, the number of holograms that are available for processing, and the center frequencies at which the holograms fall. This information is contained in a file "header", a block of bytes that precedes the data values in the data file. This header is fully described in Appendix B. The format of the data files used by GENAH is described in Appendix A. These parameters serve as guidelines; they can be altered before processing proceeds.

Many decisions need to be made in processing holograms. These options are provided as flags or additional parameters which are set by the user during the preamble. They include signal processing capabilities such as windowing, filtering, and zero–padding, as well as operational parameters such as which holograms to process. The most important parameter is the Processing Option. This option determines the processing path taken by GENAH in propagating and reconstructing the data. It includes "full" reconstructions of pressure, velocity and intensity as well as partial reconstructions in the wavenumber domain and in mixed wavenumber–spatial domains. For a complete discussion on the Processing Option, see Chapter 4 in this Guide. For more information on the processing directives and their effects on processing, see Chapter 5. All of the inputs and their allowable values are listed in Appendix C.

Once the input file has been opened, its header loaded, and the processing directives set, the output files are opened and their headers are written based on the input header and the state of the processing directives. The processing is then ready to begin.

The outermost loop of GENAH is over individual holograms. Each one exists at an independent frequency bin. (Note here the implication for parallelization on shared–memory MIMD systems or Parallel Virtual Machines: each bin can be processed independently of all other bins.) Within this loop, the processing of holograms is carried out in a series of well defined steps. This overview will cover all of the steps as if a full reconstruction were requested. Other Processing Options will not use all these steps.

On each execution of the loop (a new hologram at the next frequency), the appropriate hologram is read off the disk from the input file. The first operation is the two–dimensional forward Fourier transform of the hologram data, moving it from the spatial domain to the wavenumber domain. Since the spatial coordinates over which this is done are independent, the transform is carried out as the linear combination of one–dimensional transforms, first along the axial coordinate and then along the circumferential coordinate. The transforms are executed using Radix–2 Fast Fourier Transforms (FFT's).

This has strong implications for the size of the hologram mesh: each direction must have a number of points equal to a power of two. Prior to transforming, the data can be windowed to ameliorate aperture effects (see Section 8.3). The results from the transforms are rotated in the wavenumber domain so that the zero wavenumber component resides in the center of the data array. This makes for ease of viewing if the processing stops at this step.

Next, the propagation functions are computed. This involves computing ratios of Hankel functions, requiring double–precision computation of vectors of Bessel functions. This is one of the more expensive operations in the GENAH algorithm, but there is no good way around it. The functions are frequency–dependent, so it doesn't usually help to compute them offline and store them (the disk access would probably exceed the compute time). Once computed, they are multiplied with the wavenumber domain representation of the hologram, forming an inner product that replaces the expensive spatial convolution from which the propagation process is defined (see the theoretical discussion in Chapter 7 of the Reference Manual). The data have now been "propagated" to a new surface, the core purpose of holographic processing. Within the product loop, the data can also be wavenumber–filtered. The filter cutoff can be a constant or a function of frequency. In the latter case, filter parameters are loaded from an external file before processing begins (see Appendix E). There are two propagators, one for pressure and one for normal velocity. At the end of this operation, there are now two sets of data in the wavenumber domain.

The data are then ready to be inverse–transformed back to the spatial domain. For full reconstructions (both pressure and velocity), two sets of transforms are required. The same routines and techniques are used in back–transforming as were used in forward–transforming.

With the data back in the spatial domain, the pressure and velocity can be combined via an inner product to form acoustic intensity (see Section 9.3.11 in the Reference Manual). The total radiated power is the integral of the intensity over the reconstruction surface. It is approximated in GENAH with a rectangular quadrature over the hologram aperture. The intensity field is searched for the its maximum value, and all points in the aperture that are within 70% of that maximum are listed to an ASCII report file with the total power.

Finally, the reconstructions (pressure, velocity and intensity) can be written to their respective output files (see Section 9.3.12). The loop is then repeated for the next hologram of interest.

The other Processing Options simply step out of this path at one point or another and write what could be considered to be intermediate results. These can be very valuable, particularly results in the wavenumber domain, for other types of analysis. These possibilities are discussed throughout the Reference Manual.

3.2 A Sample Session with GENAH

This section shows an actual interactive session with GENAH . It uses the inputs contained in the file `Sample.script`. The user can compare the output listings which follow the section with an actual run of GENAH for the purposes of testing the GENAH installation (see Section 2.3).

First, the full session will be listed out as it would appear on the user's terminal. Then sections of the session will be repeated with brief commentary. For a full description of options and parameters, see Chapter 5 and Appendix C. For a complete discussion of processing options and the algorithmic response to parameters, see Chapter 4 and the Reference Manual.

This session with GENAH operates on 45 individual holograms from the input data file `Test.hol`. A complete normal reconstruction is performed using a single wavenumber filter. No additional options are used.

```
myhost: Genah> genah

                    ***  Welcome to GENAH  ***
          the Cylindrical Holography Processing Program
                    Version 1.0 September 1994

 Input Data Control Switches:
  Conjugate data? (1=exp(iwt)/0=exp(-iwt)): 0
  Enter binary number format (1=DEC/0=IEEE): 0

 --  --  --  --  --  --  --  --  --

 Enter the Input Filename:
Test.hol
```

```
-- -- -- -- -- -- -- -- --

Enter the type of units for the input file:
    0 = inches (English)
    1 = meters (MKS)
    2 = centimeters (cgs)
--> 0

Enter a choice for axial data window:
    0 = no window (rectangular)
    1 = Tukey with 8-point taper
    2 = Hanning (cosine squared)
    3 = Cosine
--> 1

Do you wish to ZERO-PAD data axially?
 (Doubles k-space resolution, reduces replicated source error)
 (1=Yes/0=No): 0

Frequency of first bin    =   198.36
Frequency width of bins   =    38.15
Number of bins available =    45
Enter the starting hologram #: 1
Enter the step interval (>=1): 1
Enter the # of holograms to process: 45

Enter the k-space filter parameters.
 ("rkc" = cutoff wavenumber, "alpha" = Filter slope)
 (Enter " 0, 0" for no filter;
  Enter "-1, 0" to read in a frequency-dependent filter
               from an external file;
 RKC, ALPHA: -1,0
Enter the file name for the variable filter:
Test.window

Additional Circumferential Order Filtering:
```

```
 Enter "1" to remove higher orders, "0" to include all orders: 0

-- -- -- -- -- -- -- -- --

Select an output processing type:
    0 = Normal Reconstruction (P, V, I)
    1 = K-space velocity withOUT propagation (K)
    2 = K-space velocity WITH propagation (G)
    3 = V(z, n) (NO inverse transform on Phi) (N)
    4 = P(kz, phi) (NO propagation, NO transform on Phi) (Q)
    5 = V(z, phi) for orders "n" = nmin,..., nmax
--> 0

-- -- -- -- -- -- -- -- --

 For the following target and test:
3300 Inner Shell Only, Flat End Caps
TEST HOLOGRAMS FOR GENAH V1.0 (DRHX=0.59055")
 these are the current values in the Header for:
  Axial lattice spacing   (inches) =      1.0000
  Shell radius            (inches) =      3.2520
  Standoff distance       (inches) =      0.0600
  Hologram Surface Radius (inches) =      3.3120
 Do these values need to be altered? (1=Yes/0=No): 1
 Enter the axial lattice spacing, "alat": 1.
 Enter the standoff distance, "drhx": 0.59055
 Enter the reconstruction radius, "radius": 3.25197

-- -- -- -- -- -- -- -- --

 Enter the Output File Name, using the following format:
  "holname.*xx" where "xx" = 10 x K_r(cutoff), or
  "holname.*vw" where "vw" represents a variable filter,
   and the "*" is literal in both cases:
Test.*vw

-- -- -- -- -- -- -- -- --
```

Bin #	1	Freq =	198.4	Max A & P =	8.6813E+00	-15.7	at	30	1
Bin #	2	Freq =	236.5	Max A & P =	6.6208E+01	-30.0	at	33	16
Bin #	3	Freq =	274.7	Max A & P =	9.9564E+00	159.3	at	35	33
Bin #	4	Freq =	312.8	Max A & P =	7.4949E+00	163.0	at	35	33
Bin #	5	Freq =	351.0	Max A & P =	6.1108E+00	163.7	at	38	34
Bin #	6	Freq =	389.1	Max A & P =	5.9833E+00	167.5	at	40	32
Bin #	7	Freq =	427.2	Max A & P =	6.3682E+00	-12.1	at	38	51
Bin #	8	Freq =	465.4	Max A & P =	9.6915E+00	170.4	at	38	1
Bin #	9	Freq =	503.5	Max A & P =	1.8090E+01	176.0	at	39	64
Bin #	10	Freq =	541.7	Max A & P =	6.2974E+01	107.5	at	25	1
Bin #	11	Freq =	579.8	Max A & P =	4.0726E+01	-20.8	at	39	1
Bin #	12	Freq =	618.0	Max A & P =	3.2896E+01	-14.9	at	40	64
Bin #	13	Freq =	656.1	Max A & P =	5.4091E+01	161.4	at	24	33
Bin #	14	Freq =	694.3	Max A & P =	4.8595E+01	-140.6	at	33	4
Bin #	15	Freq =	732.4	Max A & P =	2.6263E+01	162.0	at	26	33
Bin #	16	Freq =	770.6	Max A & P =	2.1198E+01	166.2	at	24	33
Bin #	17	Freq =	808.7	Max A & P =	2.2897E+01	167.0	at	23	33
Bin #	18	Freq =	846.9	Max A & P =	5.1654E+01	-164.1	at	22	33
Bin #	19	Freq =	885.0	Max A & P =	2.3355E+01	-16.2	at	33	1
Bin #	20	Freq =	923.2	Max A & P =	2.0145E+01	-10.0	at	33	1
Bin #	21	Freq =	961.3	Max A & P =	2.1649E+01	-0.5	at	33	1
Bin #	22	Freq =	999.5	Max A & P =	2.6507E+01	19.6	at	33	1
Bin #	23	Freq =	1037.6	Max A & P =	2.7737E+01	-39.5	at	31	49
Bin #	24	Freq =	1075.7	Max A & P =	3.6167E+01	-19.9	at	30	17
Bin #	25	Freq =	1113.9	Max A & P =	1.6432E+02	167.4	at	31	57
Bin #	26	Freq =	1152.0	Max A & P =	1.8496E+02	178.6	at	38	33
Bin #	27	Freq =	1190.2	Max A & P =	9.4159E+01	126.6	at	28	33
Bin #	28	Freq =	1228.3	Max A & P =	6.4703E+01	163.6	at	21	1
Bin #	29	Freq =	1266.5	Max A & P =	5.6077E+01	-12.5	at	44	1
Bin #	30	Freq =	1304.6	Max A & P =	4.7581E+01	-6.5	at	44	1
Bin #	31	Freq =	1342.8	Max A & P =	5.2260E+01	4.2	at	44	1
Bin #	32	Freq =	1380.9	Max A & P =	7.7005E+01	6.7	at	44	1
Bin #	33	Freq =	1419.1	Max A & P =	8.7289E+01	163.1	at	45	49
Bin #	34	Freq =	1457.2	Max A & P =	4.5445E+01	-177.1	at	45	50
Bin #	35	Freq =	1495.4	Max A & P =	4.4650E+01	-162.4	at	45	50
Bin #	36	Freq =	1533.5	Max A & P =	5.0908E+01	-141.3	at	45	51

```
 Bin #   37 | Freq =    1571.7 | Max A & P =  6.0880E+01   92.5 | at  27  34
 Bin #   38 | Freq =    1609.8 | Max A & P =  6.4391E+01  148.1 | at  45   2
 Bin #   39 | Freq =    1647.9 | Max A & P =  7.4035E+01  148.2 | at  26  33
 Bin #   40 | Freq =    1686.1 | Max A & P =  5.1020E+01  156.6 | at  26  33
 Bin #   41 | Freq =    1724.2 | Max A & P =  4.1062E+01  157.8 | at  26  33
 Bin #   42 | Freq =    1762.4 | Max A & P =  3.8541E+01  162.0 | at  25  33
 Bin #   43 | Freq =    1800.5 | Max A & P =  3.9626E+01  165.9 | at  29  27
 Bin #   44 | Freq =    1838.7 | Max A & P =  5.9036E+01  169.3 | at  29  27
 Bin #   45 | Freq =    1876.8 | Max A & P =  2.0275E+02  175.3 | at  29  27
myhost: Genah>
```

We can now take a closer look at this session. Before the file can be opened, the software needs to know what sense of time was used in creating the hologram reference file and in which number format the data is stored. Then the filename can be entered and the file opened. The power calculations at the end of GENAH require knowledge of the engineering units in which the data (pressure per unit force) are stored, so this too is entered:

```
 Input Data Control Switches:
  Conjugate data? (1=exp(iwt)/0=exp(-iwt)): 0
  Enter binary number format (1=DEC/0=IEEE): 0

 Enter the Input Filename:
./Test.hol

 Enter the type of units for the input file:
     0 = inches (English)
     1 = meters (MKS)
     2 = centimeters (cgs)
 --> 0
```

A series of questions establishes guidelines for processing the data. These include zero-padding the input data before processing, the actual holograms to be processed, the wavenumber filter to be used, and whether or not reconstructions should be restricted to a subset of available circumferential harmonics. Note that a frequency-dependent filter is invoked.

```
 Do you wish to ZERO-PAD data axially?
```

```
 (Doubles k-space resolution, reduces replicated source error)
 (1=Yes/0=No): 0

Frequency of first bin    =   198.36
Frequency width of bins   =    38.15
Number of bins available =    45
Enter the starting hologram #: 1
Enter the step interval (>=1): 1
Enter the # of holograms to process: 45

Enter the k-space filter parameters.
 ("rkc" = cutoff wavenumber, "alpha" = Filter slope)
 (Enter " 0, 0" for no filter;
  Enter "-1, 0" to read in a frequency-dependent filter
                 from an external file;
 RKC, ALPHA: -1,0
Enter the file name for the variable filter:
Test.window

Additional Circumferential Order Filtering:
 Enter "1" to remove higher orders, "0" to include all orders: 0
```

The user must choose the processing path (and consequently, the type of output) used by GENAH . Here, a full reconstruction, including pressure, normal velocity and intensity, is requested. Then the user is asked to review the critical input parameters **alat** (the axial lattice spacing), **radius** (the shell radius), and **drhx** (the radial distance to the reconstruction surface from the hologram surface, or "standoff"). Since the standoff parameter **drhx** appears to have been corrupted in the header, it is corrected here. Then a reconstruction from the hologram surface to the shell's surface can be completed.

```
Select an output processing type:
    0 = Normal Reconstruction (P, V, I)
    1 = K-space velocity withOUT propagation (K)
    2 = K-space velocity WITH propagation (G)
    3 = V(z, n) (NO inverse transform on Phi) (N)
```

```
    4 = P(kz, phi) (NO propagation, NO transform on Phi) (Q)
    5 = V(z, phi) for orders "n" = nmin,..., nmax
 --> 0

 For the following target and test:
3300 Inner Shell Only, Flat End Caps
TEST HOLOGRAMS FOR GENAH V1.0 (DRHX=0.59055")
 these are the current values in the Header for:
  Axial lattice spacing   (inches) =      1.0000
  Shell radius            (inches) =      3.2520
  Standoff distance       (inches) =      0.0600
  Hologram Surface Radius (inches) =      3.3120
 Do these values need to be altered? (1=Yes/0=No): 1
 Enter the axial lattice spacing, "alat": 1.
 Enter the standoff distance, "drhx": 0.59055
 Enter the reconstruction radius, "radius": 3.25197
```

Finally, the user must enter an output filename *template*, which the software uses to open the necessary output files. Notice the use of the extension `.*vw`. This indicates the use of a frequency–dependent filter, while the `*` is used as a marker to substitute the correct extension modifier for each output file type.

```
 Enter the Output File Name, using the following format:
  "holname.*xx" where "xx" = 10 x K_r(cutoff), or
  "holname.*vw" where "vw" represents a variable filter,
   and the "*" is literal in both cases:
Test.*vw
```

At this point, processing begins. As each hologram is loaded, it is scanned for its maximum modulus, the phase at that point, and the coordinates in the lattice where this occurs. These are typed to the screen; they are also written to the ASCII record file (with the "`.0vw`" filename extension). They are not repeated here; refer to the output listing at the beginning of this section.

When finished, GENAH will have written four files for this processing option. `Test.0vw` will contain an ASCII description of the run, with the input file maxima and output file statistics related to the maximum intensity and power in each hologram. `Test.Pvw` and `Test.Vvw` will contain the pressure and normal (radial) velocity reconstructions, respectively, in the hologram binary format. `Test.Ivw` will contain the intensity reconstructions in the hologram binary format. These files are now ready to be used in visualization and further analysis.

Chapter 4

Processing Options

4.1 The GENAH Process Path

One of the powerful aspects of GENAH is that it is built up from a set of linear operators. Processing holographic data with these operators can be likened to traversing a path along the building blocks of the algorithm, as shown in Figure 4.1. Likewise, since each dimension of holographic data is independent (two spatial and one time), the Fourier transform operations on them can be mixed in different orders. More importantly, the result of any particular traversal of the operations path has an interpretation which can bring physical information about the system to light.

The choice of which path to traverse in GENAH is, therefore, the most critical one in processing holographic data. The sections of this chapter are concerned with an introduction to the paths currently available in the GENAH code. For more detailed information about the nature and impact of these paths, see the Reference Manual. It is important that the user understand that other path choices could be coded. The ones currently available have proven to be the most used at NRL to date; the first three choices constitute 99% of all the GENAH processing done at NRL. Also, rather than pile every conceivable option into the GENAH code, other independent programs have been developed to take results from GENAH and perform other operations on them. In one important example, data are returned to the time domain from the frequency data in GENAH output to make animations of impulse responses. Similarly, the conversion of digitized time histories into

Operation	*Option 0*	*Option 1*	*Option 2*	*Option 3*	*Option 4*	*Option 5*
Fwd FT(z)	Yes	Yes	Yes	Yes	Yes	Yes
Fwd FT(ϕ)	Yes	Yes	Yes	Yes	No	Yes
Propagate P	Yes	No	Yes	Yes	No	Yes
Propagate V	Yes	No	Yes	Yes	No	Yes
Inv FT(k_z) P	Yes	No	No	Yes	No	No
Inv FT(k_z) V	Yes	No	No	Yes	No	Yes
Inv FT(n) P	Yes	No	No	No	No	No
Inv FT(n) V	Yes	No	No	No	No	Yes*
Result	$P, V, I(z, \phi)$	$p(k_z, n)$	$V(k_z, n)$	$V(z, n)$	$p(k_z, \phi)$	$V(z, \phi')$

Table 4.1: GENAH Processing Options Summary (* limited n)

frequency holograms is carried out prior to running data through GENAH , though it could conceivably be done within GENAH . In any case, the options currently available in GENAH are by no means exhaustive and could certainly be augmented.

The processing options are summarized in Table 4.1.

4.2 Option 0: Normal Reconstruction

This is the "workhorse" option in GENAH ; it represents 90% of the use of GENAH code. In "Normal Reconstruction", the full processing path of Figure 4.1 is traversed. Pressure holograms are transformed to the wavenumber domain, filtered, propagated to a new surface (and converted to velocity) and inverse transformed to spatial variables. The result is a set of pressure, normal velocity, and acoustic intensity holographic reconstructions at a surface concentric to, but different from, the original. With the intensity available, the total power in the aperture is computed. Locations in the spatial mesh which are within a specified percentage (currently 70%) of the maximum intensity are highlighted, and the relative phases of the pressures and velocities at these points give indications of how reactive the sound field is locally.

This process can involve the use of a spatial tapering window in the axial direction to mitigate truncation effects in the data aperture. It can require zero-padding in the axial direction, either to pad out to the nearest power of two number of points (for the Radix–2 Fast Fourier Transform routine used in GENAH) or to reduce aliasing from the virtual replicated sources that arise from the use of a Discrete Fourier Transform. It almost always requires the application of a wavenumber filter to reduce the high wavenumber noise inherent in backpropagation: this filter may need to change as a function of frequency. All of these are presented as choices by the software when a normal reconstruction is requested.

The key parameter in reconstruction is the radial distance between the original hologram surface and the reconstruction surface. Since GENAH has always been used to project data onto cylindrical shells, this distance is usually set as the "standoff" between the data acquisition probe (hydrophone) and the outer radius of the shell. This parameter must be set properly and checked in order to correctly reconstruct holograms backward to the shell.

4.3 Option 1: $p(k_z, n)$ *without* propagation

As mentioned above in Section 4.2, backpropagated reconstructions almost always require wavenumber filtering. The question arises, "How do I choose the filter parameters?" Most importantly, for the circular (2-dimensional) filter provided in GENAH , where should the cutoffs be placed? The quickest way to answer this question is run GENAH on a small, representative set of holograms using this Processing Option 1. This option simply forward transforms the data to the wavenumber domain (or *k-space*, as it is commonly known) without propagating to another surface. When the results are visualized (mosaics of false-color plots are most useful), the wavenumber regions containing meaningful information are usually clearly separated from higher wavenumber noise components. The gap between the two is a good place to locate the filter cutoff.

Unpropagated k-space reconstructions are also good at diagnosing any potential problems in the data. These might include aliasing, which then requires zero-padding, and tranctation effects, indicating the need for pre-windowing the data spatially in the axial direction. When plotted in the frequency-wavenumber domain as a function of circumferential order, dispersion surfaces reveal which wavetypes exist in the nearfield of the shell.

4.4 Option 2: $V(k_z, n)$ *with* propagation

When a filter has been selected, it is best tested in the wavenumber domain. This is accomplished using this processing option, which, in addition to the forward transform of the previous section, applies the velocity propagator. The results, when visualized, will clearly reveal whether the wavenumber filter cutoff(s) have been properly set. In this way, this processing option provides another diagnostic step along the road to full reconstruction.

However, it is also extremely important in and of itself in generating frequency-wavenumber plots (dispersion surfaces), as mentioned above. Since these data have been *propagated* (usually to the shell surface), they represent the actual normal velocity wavenumber content of the shell itself, a key component in diagnosing shell physics. Thus, output from this processing option is critically important to many analyses of shell vibrations. Further wavenumber filtering can be developed to limit the data to left- or right-traveling waves only or to exclude subsonic or supersonic waves. The results of this Processing Option 2 are critically important in assessing the application of further operations.

4.5 Option 3: $V(z, n)$ (no inverse transform on ϕ)

This is the first of the specialty options in the GENAH processing path. After having been converted to velocity and propagated to the new surface, the data are inverse transformed only in the axial direction. This results in reconstructions which are still separated as a function of circumferential order. In effect, the routine just bypasses the transform back to angle. This is valuable in examining wave activity specific to certain orders. Many scattering and radiation phenomena are likewise classified by these circumferential orders. The lowest orders (0, 1 and 2) are often of greater importance in the study of shell vibrations, since they contribute most of the power to the acoustic farfield.

Processing Option 3 is also valuable in comparing with the results of many available Finite/Infinite Element codes for axisymmetric bodies. These codes divide their processing across the independent orders as well. In some cases, it is more important to compare the synthetic models to the experimental data order by order rather than having them summed. Option 3 enables these comparisons.

4.6 Option 4: $p(k_z, \phi)$ (no propagation, no transform on ϕ)

This is the second of the specialty options in the GENAH processing path. It is not one of the most useful, but it is definitely the simplest. The result of this option is simply to forward transform the pressure holograms in the axial direction only. The results are then written as a function of angle at the original hologram surface, that is, $P(k_z, \phi)$. No propagation or wavenumber filtering is performed.

Option 4 can be used on non–holographic data which require one–dimensional forward Fourier Transforms. For instance, if such data were stored in a single "hologram" as a function of one spatial dimension and frequency, this option could be used to transform the spatial dimension to wavenumber without modifying the data in any other way. Thus, GENAH can be used as a tool for transforming data that is not actually holographic.

4.7 Option 5: $V_n(z, \phi')$, Limited Orders

This is the third of the specialty options in the GENAH processing path. It is the most similar to a normal reconstruction, with two exceptions. First, only velocity is reconstructed; pressure and intensity are not available. Second, the inverse transform from circumferential orders to angle is limited to a subset of the available orders as specified by the user. Unlike the filtering option which limits the orders to $(-n_{max}, \ldots, +n_{max}; 0 \text{inclusive})$, Option 5 allows the user to specify a range of orders $n = (n_{min}, \ldots, n_{max})$ inclusive. Thus, the user could effectively exclude the lowest circumferential order(s) if so desired.

However, the remaining orders are not Fourier summed into a single reconstruction at each frequency. Instead, for each order $|n|$, both the positive and negative components are summed back to an angular representation in ϕ. The reconstruction at each order $|n|$ is then written for each frequency.

Thus, if M bins are processed with N orders each, the resulting file will contain $M \times N$ reconstructions. This will impact any post–processing routines which compute a given reconstruction's frequency by multiplying the bin number by the frequency interval.

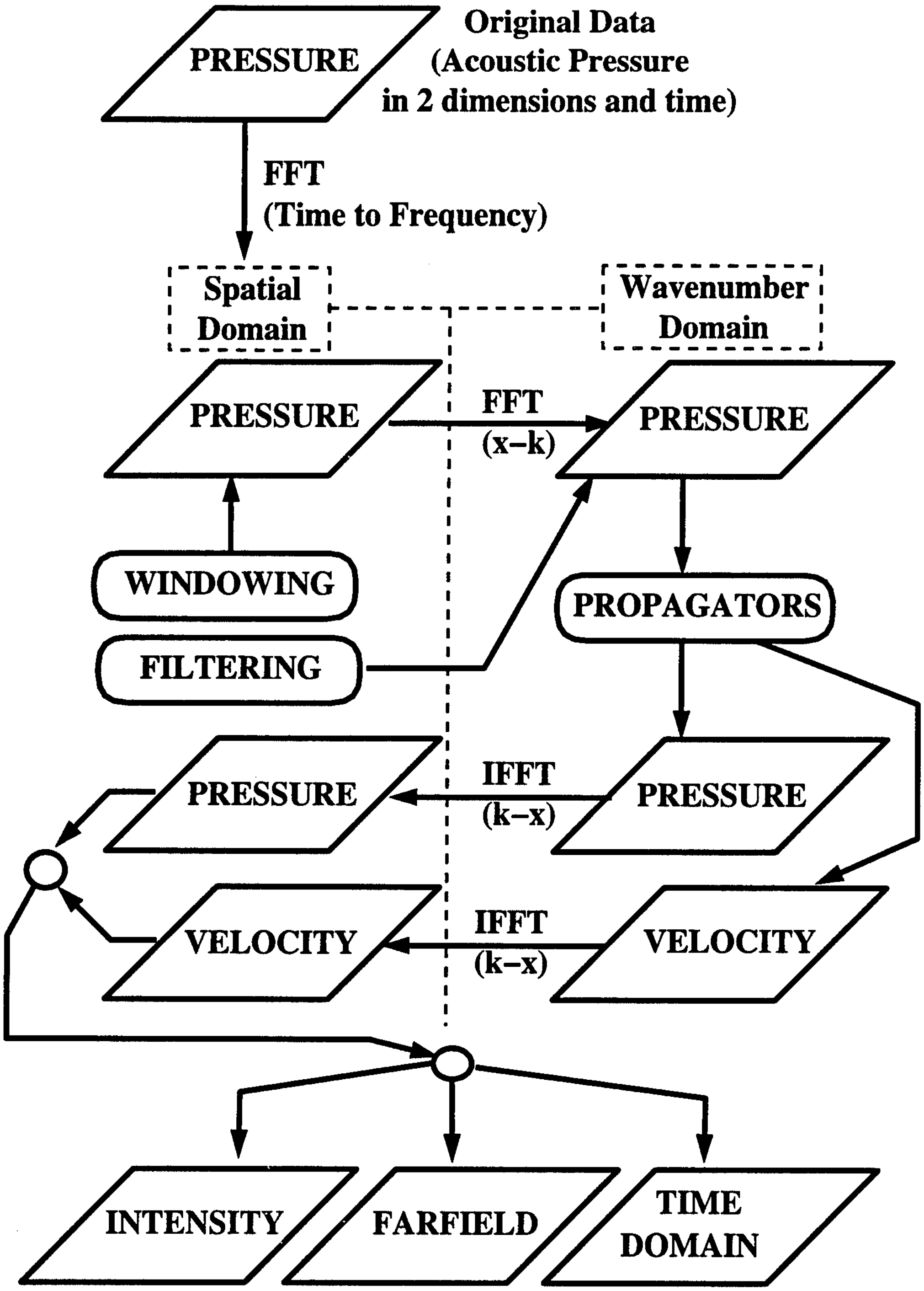

Figure 4.1: Schematic of the Nearfield Acoustical Holography processing path.

Chapter 5

Inputs and Parameters

A very brief introduction to the user inputs needed for a GENAH run was given in Chapter 3. This chapter is intended to explicate all of the possible responses to requests for inputs in GENAH and how these responses will affect the outcome of the processing. It is not intended as a tutorial on the algorithm: that information is included in Chapter 8. Instead, this chapter will simply define allowed responses and indicate how the routine will respond to the choices the user makes.

5.1 Input Data Information

Each file written in NAH format (see Appendix A) has a short header (see Appendix B) with a good deal of information about the data in the file. Nonetheless, some information must be known about the file before it can be located, opened and read. This section describes four items (and one error check) that GENAH needs to accomplish this.

5.1.1 Input Data Control Switches

The following two data type switches are included in the current version of GENAH for historical reasons.

The first switch (`iconj`) allows the user to conjugate all incoming data. This has the effect of reversing the sense of time in the expression of a propagating wave, that is, either $e^{i(\omega t-kr)}$ or $e^{-i(\omega t-kr)}$. The Fourier transforms within GENAH are fixed to recognize the first sense as an outgoing (radially) or rightward going (axially) wave. However, different preprocessing routines which generate the input holograms from digitized data may use different FFT routines with conflicting senses of time (or in this case, forward *vs.* inverse transforms). In particular, the FFT routines provided with MATLAB are known to cause this problem. The user should find out which sense of the imaginary number is used in any preprocessing. If this is not possible, it can be tested by processing a few higher frequency holograms and examining data well outside the test region for increasing phase in the near to far field transition (outward going waves). If the phase of the reconstruction is generally decreasing away from the source, then the opposite time sense must be employed. When the FFT's provided with GENAH are used in preprocessing (and the time and spatial transforms use opposite senses, respectively), then respond to the first switch with "0". Otherwise, respond with "1". Again, be sure to test any new situations to be sure.

Summary:	`iconj`	`Conjugate data?(1=exp(iwt)/0=exp(-iwt)):`
Response:	`1`	Conjugate all incoming data.
	`0`	Do not conjugate any incoming data.

When running on a machine which uses the IEEE number format (most workstations and processors), The second switch (**`ivax`**) allows GENAH to read in binary hologram data stored in DEC number format. It simply converts all incoming values under the assumption that the data are DEC binary while the executable is running under IEEE binary. Note that the converse is *not* provided as an option in this release of GENAH (*i.e.*, conversion from IEEE format to DEC format). If such a need were to arise, it would require modification of the source code. Contact the authors in that case. This switch was provided because the code, originally running on VAX machines, was ported to SGI machines, while data taken up to that time was not. Note also, that if both the data and the code are native to a DEC machine, the user would respond as if no conversion was necessary (*i.e.*, using the IEEE switch). Note that some machines, such as the Convex C-Series, have both native (IEEE) and DEC-compatibility modes. In such cases, it is advisable to compile and run GENAH in the mode in which data is stored rather than using conversions.

Summary: **`ivax`** `Enter binary number format(1=DEC/0=IEEE):`

Response:	`1`	Convert from DEC format to IEEE format.
	`0`	Do not perform any number format conversions.

5.1.2 Input Data File Name

The name of the disk file containing holographic input data (`fname`) includes the full directory path and the file's native name. Note that the string as entered *must not exceed 80 characters in length.* This can easily occur when data is being read from a remote disk and/or from deep within a directory tree. Care should be exercised in these cases. When a conflict arises, it is best to run GENAH from a directory nearer to the data (to shorten the name) or to set up an alias (Unix) or symbol (VMS) which points to the data.

Summary:	`fname`	`Enter the Input Filename:`
Response:	(Unix)	`name` *or*
		`./name` *or*
		`dir/name` *or*
		`dir/subdir/.../name` *or*
		`/dir/subdir/.../name` *or*
		`/partition/dir/subdir/.../name` etc.
	(VMS)	`name.ext` *or*
		`[dir]name.ext` *or*
		`[dir.subdir...]name.ext` *or*
		`disk:[dir.subdir...]name.ext` *or*
		`machine::disk:[dir.subdir...]name.ext` etc.

5.1.3 Engineering units for input spatial measures

Data coming into GENAH is assumed to be acoustic pressure per unit reference (a force gauge, an accelerometer, another hydrophone or the drive voltage) in SI units, *e.g.* Pascals per Newton. However, the engineering units used for spatial measure of the aperture and standoff can be in any system of units, because the propagator functions depend only on the ratio of spatial measures. (An exception to this is the specific acoustic impedance ρc, used in converting from pressure to velocity. It is fixed in SI units for water at atmospheric pressure at 20°C.) Even so, the units of spatial measure must be known to the routine for two reasons. First, wavenumber filtering requires knowledge of the axial lattice spacing to form axial wavenumbers. This impacts the values entered in for the wavenumber filter parameter `rkc` ($k_{r,cutoff}$) as well. Second, computation of total power requires integration of the intensity over the total aperture, whose area must be known.

Historically, the lattice spacing and standoff were measured exclusively in inches. More recently, these values have been stored in meters or centimeters for compatibility with other acoustic measurement programs. Since an entry was not provided in the file header (see Appendix B) for a units flag, this has been left as an input parameter.

Summary:	`units`	`Enter the type of units for the input file:`
Response:	`0`	inches (English)
	`1`	meters (MKS)
	`2`	centimeters (cgs)

5.1.4 Correction for number of points in the circumferential direction

When NAH data are taken at NRL, an extra axial scan line is always added which overlays the first scan line. In this way, data are available from the very beginning and end of the experiment which should theoretically be identical. This provides a good measure of the variation which has occured over the course of the experiment. This variation commonly arises from water temperature changes or amplifier level drifting. The number of axial scan lines surrounding the source, *i.e.* the number of lattice points in the circumferential direction (`ncirpts`), is always chosen to be even and almost always a power of two for the radix-2 FFT in GENAH . The extra line is counted in before preprocessing. It can appear in the original data header (see B as an odd number of points in the circumferential direction, and there have been instances in which this is not corrected in generating the input hologram file.

Since GENAH depends on this number for all of its operations, especially the FFTs, it *must* be even. (The only exception is in the event of a single scan line, in which case the data are replicated to form a synthetic, axisymmetric field.) In the event that this value is not an even number when GENAH opens the header field for the input file, an error is flagged, and the user is given the opportunity to correct the value. Note that the actual number of data lines in each hologram must agree with the corrected value; otherwise, the algorithm for counting records in the file will fail. Again, this input will only be requested if GENAH detects an odd number in the header location. The user must know if this situation will occur before writing a shell script to execute GENAH or the subsequent inputs will be out of order and cause the routine to fail.

Summary:	`ncirpts`	`*** ERROR *** in "ncirpts"`
		`Current value is NOT acceptable.`
		`Please reenter "ncirpts":`

Response:	"n" (even)	Actual number of scan lines in file (must be even, power of 2)

5.2 Processing directives

This section describes seven parameters which control the processing done by GENAH . These include spatial windowing and padding, choosing which holograms (frequency bins) to process, wavenumber-space filtering, processing path option and reconstruction parameters.

5.2.1 Axial direction spatial window

Hologram measurement apertures are usually designed to extend sufficiently past the ends of the source that the acoustic pressure at the edge of the aperture has dropped below the measurement signal to noise ratio (SNR). This is referred to as "overscan". For more details regarding this, see Section 8.3 in the Reference Manual. For many measurements, this cannot quite be achieved, either because full overscan would not allow a sufficient number of points to fall over the source itself, or because at certain frequencies the source tends to project much more energy axially than over the rest of the frequency range of interest. For whatever reason, any significant pressure level remaining at the ends of the aperture will appear as a truncation discontinuity to the Fourier Transforms. This can strongly bias the results of a reconstruction. To avoid this, four tapering windows (one of which is simply the rectangular window) have been made available in GENAH to force the data to zero at the ends of the aperture.

The first choice is simply not to taper the data at all. This rectangular or "box-car" window is fine if the overscan is sufficient at all frequencies.

It yields the narrowest main lobes possible in the wavenumber domain, but with the highest sidelobes.

The second choice is the workhorse for most NRL processing. This is the Tukey window. It has no taper over most of the aperture (that is, a value of 1.0), then a short cosine taper over the last eight points at each end. This yields wavenumber responses almost identical to those of the rectangular window for apertures of 128 or more points while eliminating truncation problems.

The third and fourth choices are cosine tapers. The Hanning, or cosine-squared taper is the most severe of all the windows. The Hamming, or cosine window is less severe. Both windows affect the entire aperture, with a value of 1.0 only at the center point. The Hanning window provides one more continuous derivative at the boundary than the Hamming, yielding faster sidelobe roll-off in the wavenumber domain. These windows must be used with caution. They tend to bias the reconstructions strongly (they leave their footprint, so to speak, on the results). They are most useful in reducing background in the wavenumber domain; this is critical in trying to find weak trends in visualizations of that domain. Their use at NRL has been confined to sources with very strong endfire and in scattering holography with bow incident wave energy.

Note that adding windows of other types to the GENAH code would be straightforward. The user would simply add it to the list of choices and place the appropriate subroutine call into the `if ... then` tree which loads the windowing vector, `tuk`.

Summary:	`window_choice`	`Enter a choice for axial data window:`
Response:	0	No window (Rectangular)
	1	Tukey with 8-point taper
	2	Hanning (cosine squared)
	3	Hamming (cosine)

5.2.2 Axial direction zero-padding

Much of the art of processing nearfield holograms lies in taming the artifacts introduced by the use of discrete, finite length Fourier transforms. One powerful weapon in the user's arsenal is the ability to zero-pad the input stream. This has two valuable effects. First, it doubles the distance between the true source and its replicated images, greatly reducing the effects of overlap aliasing (*i.e.*, the energy from the image sources is less likely to leak into the source aperture). Second, padding in the spatial domain has the effect of interpolating in the wavenumber domain. While no true additional information is introduced (the interpolated points fall on $sin(x)/x$ curves whose peaks occur at the original wavenumber values), the interpolation is very helpful in making plots of wavenumber response.

For these reasons, GENAH includes an option to zero-pad the input data. The option should be used with care, as it increases processing time and memory usage. Output holograms in the wavenumber domains (Processing Options 1 and 2) will be twice as large as the input holograms. Note that reconstructions back to the spatial domain do not retain the extra points; they are written out using the same number of axial lattice points as the input files.

Summary:	`izadd`	`Do you wish to ZERO-PAD data axially?`
Response:	1	Yes, zero-pad input data
	0	No, do not zero-pad input data

5.2.3 Hologram frequency bin control

It is not always the case that the user wishes to process every available hologram in an input file. A set of controls is made available to select the sequence of a subset of the holograms. These include the bin number at which the sequence begins, the step interval (*i.e.*, the number of bins to skip between holograms, including the next to be processed), and the total number of bins to process in this run. It is worthwhile to insure familiarity with this method of enumerating which holograms are to be processed from the original set. An example case will make this more clear.

Suppose 1000 holograms are available in the original data set. The first hologram is associated with the frequency bin centered at 1000 Hz, and each bin has a width of 10 Hz. This means the last available bin will be centered at 10990 Hz. If the user wishes to process all of these holograms, the starting hologram will be "1" (even though it is not at the 0-frequency bin), the step interval will be "1" (incrementing to each successive bin) and the total number of holograms to process would be "1000". If, however, the user needs information only on the range from 2000 to 3000 Hz in 100 Hz increments, the starting hologram would be "101", the step interval would be "10" and the number of holograms to process would be "101".

The most important distinction here is that one does not enter the first and last holograms to process, but the *total number* of holograms to be processed. Also of importance is the fact that if any processing were to be done on the resulting (processed) data, the bin at 2000 Hz would become the first bin of that data, and a step interval of 1 would span 100 Hz. In other words, hologram counting is always local to the file being processed; there is no "memory" of which bins were selected from a previous input file.

Summary:	`ihstart`	`Enter the starting hologram #:`
Response:	"ℓ"	The number of the first frequency bin to be processed from the available set

Summary:	`ihstep`	`Enter the step interval (>=1):`
Response:	"m"	The increment to the next successive frequency bin to be processed from the available set

Summary:	`ihno`	`Enter the # of holograms to process:`
Response:	"n"	The total number of frequency bins to be processed from the available set

5.2.4 Wavenumber filter control parameters

Wavenumber filtering is a key task in nearfield acoustical holography processing. It is perhaps the one aspect of the processing that continues to require human intervention, eluding fully automated or "turn-key" operation. Of all the parameters available to the holographer, it is the filters that most influence the outcome of reconstruction. The need for filtering and the theoretical underpinning for it are more fully explored in Chapter 7 of the Reference Manual. The process of actually selecting the cutoff and slope parameters are described in Appendix E. This section confines itself to the mechanics of entering the filter parameters into the GENAH algorithm.

Only a single type of filter is available in the current implementation of GENAH . It is a flat-top filter with a matched exponential taper at the edge. Researchers have explored various other filters, but no particular benefits were established. The current filter is well understood, robust and has withstood the test of time.

The filter is computed live within GENAH , and at any given frequency only two parameters are needed. The first, k_{cutoff}, is the wavenumber at which the taper, or "cutoff", begins. This value is measured in radians per unit length, or wavenumber. The axial wavenumber is computed as $k_z = 2\pi/\lambda_z$, and the circumferential wavenumber is $k_\phi = n/r$. (Note that while k_z is continuous, k_ϕ exists only at the integer values of n. However, k_z becomes discretized in practice when the aperture is sampled. See Chapter 8 for more information.) This requires that the user know what units are being used to measure axial lattice point spacing and the radius (see Section 5.1.3 above). The second parameter, α, controls the slope, or rate of taper, of the exponential tail. This parameter will be explored further in Appendix E; for now, it is sufficient to state that $\alpha \approx 0.05$ is acceptable.

Since the region of useful wavenumbers tends to grow with frequency, a single filter setting may not be adequate for the entire range of frequencies to be processed. In this case, an option is available to read in a set of filters from an external ASCII file. Again, for details, consult Appendix E. Also, in order to explore where the cutoff should be placed, a preliminary reconstruction of velocity wavenumbers with no filtering may be needed. An option is available to apply no filter at all.

Summary:	`rkc, alpha`	`Enter the k-space filter parameters`
Response:	"0, 0"	Do not apply any wavenumber filter
	"-1, 0"	Read in a frequency-dependent filter from an external file
	"a, b"	$k_{r(cutoff)} = a$ radians per length, Slope $= b$

If the user chooses to read in a frequency independent filter set from an external file, GENAH immediately prompts for the name of that file.

Summary:	`wname`	`Enter the file name for the variable filter:`
Response:	"`name.ext`"	The full path and file name of the file which contains the parameters for the filter set. (Limited to 80 characters or less.)

There is not much error checking on the filter parameters. If, however, a filter set has more filter points specified than can be stored in the available array, the following error will be announced:

```
*** FATAL ERROR *** in Variable Filter:
 Number of entries exceeds size of "wbin".
 Increase parameter "nrkcbins" and recompile to continue.
```

The program will be terminated immediately. The user should then edit the parameter "`nrkcbins`" in the GENAH code, increasing it enough to accomodate the larger filter. In this release of GENAH , `nrkcbins` has been set to a value of 250.

For cylindrical holograms, the circumferential wavenumbers, or orders, are independent of one another. They may be additionally filtered (removed completely) without impacting other orders or their individual contributions to the reconstructions. An extra option is provided in GENAH to remove (zero) any orders above a chosen maximum, in addition to any filtering in the wavenumber domain. Order filtering is available for all Processing Options except Option 5 (see Chapter 4), for which separate order filtering is available. Note that this general order filtering is low-pass only, *i.e.*, all orders from $-n_{max}$ to $+n_{max}$ (including 0) are automatically included.

Summary:	`nfilt`	`Additional Circumferential Order Filtering:`
Response:	0	Include all available circumferential orders (No filtering)
	1	Remove circumferential orders above a set maximum

If a response of "1" is given, then GENAH further requests the (integer) maximum circumferential order to be used in reconstructions. This number must be less than or equal to one half the total number of circumferential lattice points (or axial scan lines). For example, if there are 64 circumferential data locations, then the available orders would include $-32, \ldots, +31$, inclusive. It would not make sense to request orders higher than 32 to be removed in this circumstance.

Summary:	`nf_max`	`Enter the maximum order "n" to INCLUDE`
Response:	"n_{max}"	The maximum circumferential order included in reconstruction ($\leq$ `ncirpts`/2).

5.2.5 GENAH processing path option

The GENAH processing path is the key option in choosing a reconstruction result type. The nature of these paths was discussed in detail in Chapter 4; those details will not be revisited here. This section is included for completeness and as a reference aid for the user. In the table that follows, the capital letter following each entry is the identifier added to the filename extension when the output file is opened.

Summary:	`ifqout`	`Select an output processing type:`
Response:	0	Normal Reconstruction (P, V, I)
	1	K-space velocity *without* propagation (K)
	2	K-space velocity *with* propagation (G)
	3	$V(z,n)$ (NO inverse transform on ϕ) (N)
	4	$P(k_z,\phi)$ (*no* propagation, *no* transform on ϕ) (Q)
	5	$V(z,\phi')$ for orders $n = n_{min},\ldots,n_{max}$.

If Processing Option 5 is chosen, this is the point at which GENAH will request the actual range of circumferential orders to be included. Unlike the general circumferential order filtering mentioned in the previous Section which is low-pass only, both a minimum and a maximum may be specified with Option 5. However, only the reconstructed velocity will be available as an output. Note that the specified minimum and maximum apply to *both* positive and negative orders. For example, in a scan with 64 circumferential lattice points, orders would include $-32,\ldots,+31$, inclusive. If the Option 5 filter was set at "1,3", then the actual orders used in reconstruction would include $(\pm1,\pm2,\pm3)$. This would remove the $n=0$ "breathing mode" while limiting reconstruction to the (usually more important) modes below the fourth.

Summary:	`nmin, nmax`	`Enter the minimum and maximum circumferential orders to keep in processing:`
Response:	"n_{min}, n_{max}"	Lowest and highest orders to retain ($0 \le n_{min}, n_{max} \le$ `ncirpts`/2.)

5.2.6 Processing parameter adjustment

There are several key spatial measurements without which holographic reconstructions cannot proceed. The parameters include the axial lattice spacing, the shell radius and the standoff distance from the shell radius to the hologram measurement surface. These parameters are carried in the hologram file header (see Appendix B). However, from time to time they can become corrupted prior to processing. They are of sufficient importance that GENAH always lists them for the user prior to proceeding with reconstructions, and offers the user the opportunity to alter or correct them.

One particular case in which they might be altered even though they are correct is when the user wishes to reconstruct to some surface other than the outer radius of the shell. A typical example would be when the normal velocity and intensity are needed at the hologram surface itself (a propagation distance of zero). In that case, the shell radius would be increased to be equal to the hologram surface radius and the standoff distance would be reduced to zero.

Note that the hologram surface radius is a derived quantity, computed by adding the shell radius and the standoff distance. This is due to the fact that the latter two are measurable quantities. Nonetheless, for reconstruction purposes, the hologram surface radius is the radius from which reconstructions proceed. If the reconstruction radius needs to be altered from the shell surface, be sure that the sum of that new radius and the new standoff distance are equal to the original hologram surface radius. Otherwise, the results will be biased. As an example, suppose a 5 inch radius shell is measured with a 1 inch standoff. The parameters would be announced as follows:

```
these are the current values in the Header for:
 Axial lattice spacing   (inches) =     1.0000
 Shell radius            (inches) =     5.0000
 Standoff distance       (inches) =     1.0000
 Hologram Surface Radius (inches) =     6.0000
```

For these conditions, the hologram surface radius is reported as 6 inches, the sum of the shell radius and the standoff distance. Suppose the user wished to reconstruct only halfway in, say to the surface of a half-inch thick test tile. Then the shell radius would be increased to 5.5 inches and the standoff distance would be reduced to 0.5 inches. The hologram surface radius would remain at 6.0 inches, but GENAH would only reconstruct inward 0.5 inches.

This unfortunate detail of the GENAH code is an historical one that is not likely to be altered in the future.

Summary:	`ichng`	`Do these values need to be altered?`
Response:	0	The reported parameters are correct: Proceed with reconstruction.
	1	The reported parameters need to be altered.

For any non-zero response to the parameter change query, the following set of requests are made:

Summary:	`alat`	`Enter the axial lattice spacing, "alat":`
Response:	"a"	Axial lattice spacing in units "`units`"

Summary:	`drhx`	`Enter the standoff distance, "drhx":`
Response:	"b"	The distance, in units "`units`", from the hologram surface to the desired reconstruction surface.

Summary:	`radius`	`Enter the reconstruction radius:`
Response:	"c"	The radius of the desired reconstruction surface, in units "`units`"

5.3 Output file name

When an algorithm is constantly repeated or reused, the issue of filenaming is never a simple one. GENAH gives almost complete flexibility in naming files with two caveats. First, the name, including the full path, must be less than 80 characters (see Section 5.1.2 for examples in Unix and VMS). Second, GENAH does try to control the three–character extension to the filename. This is most effective in normal reconstructions (Processing Option 0), where four output files are created. The same path and file name are used, but the extensions begin with different letters depending on the data contained in the file. These letters and their corresponding file types are listed below in Table 5.1. The last two letters of the extension are either two numerals or the letters "vw". If a fixed wavenumber filter is used for the entire reconstruction set, the two numerals are the cutoff wavenumber multiplied by 10. For example, if $k_{r(cutoff)} = 6.0$, the extension numerals would be "60". If, on the other hand, a frequency-dependent filter file was used, the letters "vw" are appended as an acronym for "*v*ariable *w*indow".

Extension	*Data Type*
`O`	ASCII processing description output file
`P`	Spatial domain Pressure, $P(z, \phi; r')$
`V`	Spatial domain Velocity, $V(z, \phi; r')$
	Order-limited Velocity, $V(z, \phi'; r')$ $(n = n_{min}, \ldots, n_{max})$
`I`	Spatial domain Intensity, $I(z, \phi; r')$
`K`	Wavenumber domain Pressure *without* propagation, $p(k_z, n; r_0)$
`G`	Wavenumber domain Velocity *with* propagation, $v(k_z, n; r')$
`N`	Axially reconstructed Velocity, $V(z, n; r')$
`Q`	Axially transformed Pressure, $P(k_z, \phi; r_0)$

Table 5.1: First letter of filename extensions for GENAH output files.

Summary:	`fname1`	`Enter the Output File Name:`
Response:	`name.*mn`	$mn = 10m.n = 10k_{r(cutoff)}$
	`name.*vw`	"vw" for *v*ariable *w*indow (frequency-dependent wavenumber filter)
		(Note that the character "*" is literal here)

Chapter 6

Input and Output Files

Perhaps the most cumbersome hurdles in the use of GENAH are the format and structure of the data files for input and output. Again, it is the historical evolution of the code which led to the current situation.

When it was originally developed on a PDP-11 platform for planar holography, the available storage (the 8-inch floppy) was quite dear, and the data needed to be stored as compactly as possible. This required the use of binary storage (4 bytes per real number, 8 bytes per complex number). When the broadband system was introduced, analyses required being able to access any particular hologram immediately, without having to step through the entire file each time. This required the direct access mode of storage. Direct access modes in `FORTRAN-77` require a fixed record length. This immediately led to a conflict, in that different size holograms would demand different record lengths, and not all computer systems store record length information externally. Since almost all holograms at that time had meshes of 64 by 64 points, it was decided that each scan line should constitute a record. 64 complex values require 512 bytes of storage, and this was chosen as the record length for all subsequent holograms.

Thus, every hologram taken since 1985 at NRL is stored in direct access mode with a fixed record length of 128 longwords (4 bytes per longword). While complicating matters for different hologram sizes, this has made all previous experiments accessible to current software without modification. Also, both input and output files for every stage of processing share this same structure and format, so the same I/O software is reusable at every stage of processing. The format of these files is defined formally in Appendix

A. It is discussed here briefly so that the user can proceed with reading and writing holograms.

A hologram is composed of a set of scan lines. For scan lines with $M = 2, 4, 8, 16$ or 32 axial locations, $N = 64/M$ scans fit in each record. For $M = 64$ locations, each scan line constitutes a single data record. For scans longer than 64 locations, additional records are used for each scan line. So, for instance, a 128–point scan requires two records; a 256–point scan requires four records. In short, the number of records required for each scan is the number of points in the scan divided by 64, rounded up to the nearest integer. The current version of GENAH is limited to 256 axial locations per scan line.

An individual hologram (at one frequency) requires as many records as there are scan lines times the number of records per scan line. For example, a typical cylindrical hologram of 128 points per scan line and 64 scan lines would require 128 records of 512 bytes each, for a total of 64 kilobytes of storage. A hologram with 32 points per scan line and 32 scan lines would require 16 records of 512 bytes each, for a total of 8 kilobytes of storage.

The first record (512 bytes) of each file (whether input or output) is dedicated to a header record which contains "meta-data", *i.e.*, data about the holograms. This header is difficult to extract (again for historical reasons). It is critically important however, because it contains information on the size of the holograms, their number, and all of the pertinent information about the experiment needed by GENAH for processing. The structure and contents of the header record are discussed in complete detail in Appendix B. Since this header record is always the first record, accessing a particular hologram record requires that 1 be added to the computed count to point to a given record in a specific hologram.

It is implied above that a scan line may have any number of points. However, as currently written, GENAH strongly restricts the allowable record sizes for reading and writing holograms and reconstructions. This originally grew from the fact that the number of points needed to be a power of two for the Radix-2 FFT routine used in GENAH . Currently, GENAH will only allow input files to have 16, 32, 64, 128 or 256 points per scan line exactly. If the data contain some other number of points per scan line, they must be preprocessed and rewritten, padded out to the next highest of these values. This rather restrictive practice is currently under review, and a future release of GENAH will most likely provide much greater flexibility in the number of points per scan line. Even then, however, the data will be padded out to the next power of two for the FFT. If the user needs even more flexibility, some other FFT routine will need to be substituted for the one provided in this GENAH release. In any event, with the source code in hand, the user is free to alter the I/O to suit local needs.

The available output sizes from GENAH are also restricted. (This, too, is under review for a future release.) For spatial reconstructions (Processing Options 0, 3 and 5, see Section 4), only 16, 32, 64 and 128-point outputs are currently allowed. If the axial data have been zero-padded (`izadd` set to 1), then only the center $M/2$ points of an M–point scan line are written, so that the output line length is equal to the input line length. For wavenumber-domain outputs (Processing Options 1, 2 and 4), up to 256 points can be written out. In this case, it does not matter if the data had been zero-padded (up to a maximum of 256 points per line); all of the wavenumbers are written.

Part II

Reference Manual

Chapter 7

Theory of Nearfield Acoustical Holography

7.1 Introduction

The reconstruction of wave fields from holographic measurements has a long history in this century. It derives fundamentally from applying Green's Theorem to the Helmholtz Equation for linear wave fields. While the optical version of holography is by far more well known, applications to acoustical fields have been known for decades.

The idea that an acoustical field could be reconstructed at its source with a resolution that exceeds the acoustical wavelength limit dates to the seminal paper of Williams and Maynard in 1980 [1]. It was shown there that under certain circumstances, non-propagating (or *evanescent*) waves whose wavelengths were smaller than the acoustic wavelength at a given frequency could be reconstructed. This reveals wave patterns distributed over the source surface which do not necessarily propagate to the acoustic farfield but which are critical to an understanding of wave motion on and energy propagation through the source structure. Thus, *Nearfield Acoustical Holography* (NAH) was born.

What follows is a brief overview of the theory of NAH. For more detail, the reader is referred to the very thorough development in Maynard, Williams and Lee [2]. This work is quite complete and fleshes out many of the details glossed over here. The extension of the theory in Cartesian coordinates to cylindrical coordinates is touched upon in that paper, but it is more elaborately treated in Williams, Dardy and Washburn [3]. The development of an algorithm for computer implementation of NAH is straightforward, but the treatment of errors and other issues is thoroughly covered in Veronesi and Maynard [4], a companion article to [2]. While the literature is rich with other articles (mostly dealing with implementations of NAH), these three form a useful core for understanding nearfield holography, and all three are found in the Journal of the Acoustical Society of America, which is readily available in most technical libraries.

7.2 Analytical Development of Holography

7.2.1 An Informal Sketch of Holography

We begin our discussion of NAH theory with a simple motivation that reveals the concept behind reconstruction. This is followed by a more detailed analysis of the functions involved for Cartesian Coordinates. Then, in Section 7.4, the analysis is extended to cylindrical coordinates.

Imagine a Cartesian half-space, that is, a three dimensional space extending upward above a planar "floor" of infinite extent. Locations on the floor are denoted by coordinates x and y, and distance above the plane is measured along coordinate z. Now imagine that this half-space is filled with a fluid which will bear acoustic disturbances, but that no source of disturbance is located anywhere above the floor. The propagation of acoustic pressure waves in the half-space is described by the wave equation:

$$\nabla^2 p(x,y,z;t) - \frac{1}{c^2}\frac{\partial^2}{\partial t^2}p(x,y,z;t) = 0 \ , \qquad z > 0 \ , \tag{7.1}$$

where c is the speed of sound in the fluid. Assume the time variation of any waves in the space (or on the surface $z = 0$) is harmonic, that is, of the form $Ae^{-i\omega t}$ with some frequency ω. If we let $p(x,y,z;t) = p(x,y,z)e^{-i\omega t}$ and substitute this into 7.1, we arrive at the Helmholtz equation:

$$\nabla^2 p(x,y,z) + k^2 p(x,y,z) = 0 \ , \qquad z > 0 \ , k^2 = \left(\frac{\omega}{c}\right)^2 . \tag{7.2}$$

In essence, we have performed a Fourier transform on the wave equation to reduce the dependence on time to a set of independent equations for different (arbitrary) frequencies.

We formalize the concept of the Fourier transform by defining the transform pair for a spatial variable ξ:

$$P(k_\xi) = \int_{-\infty}^{\infty} p(\xi) e^{-ik_\xi \xi}\, d\xi \ , \tag{7.3}$$

$$p(\xi) = \frac{1}{2\pi} \int_{-\infty}^{\infty} P(k_\xi) e^{ik_\xi \xi}\, dk_\xi \ . \tag{7.4}$$

An important property of this transform is that when it operates on the second derivative of a function, the result is just the negative of the squared frequency times the transform of the function itself:

$$\int_{-\infty}^{\infty} \frac{d^2}{d\xi^2} p(\xi) e^{ik_\xi \xi}\, d\xi = -k_\xi^2 P(k_\xi).$$

The ∇^2 operator in Cartesian coordinates is simply the sum of the second derivatives in each direction.

If we apply the Fourier transform to the x and y directions of 7.2, this property of Fourier transforms yields

$$-k_x^2 P(k_x, k_y, z) - k_y^2 P(k_x, k_y, z) + \frac{d^2}{dz^2} P(k_x, k_y, z) + k^2 P(k_x, k_y, z) = 0 \ , z > 0 \ , \tag{7.5}$$

where $k = \omega/c$ is known as the propagating wavenumber in the fluid. It is related to the fluid wavelength λ at frequency ω by the equation $k = 2\pi/\lambda$. When the terms in equation 7.5 are regrouped, we obtain

$$\frac{d^2}{dz^2} P(k_x, k_y, z) + [k^2 - (k_x^2 + k_y^2)] P(k_x, k_y, z) = 0 \ , \qquad z > 0 \ . \tag{7.6}$$

This expression is a simple, second order, ordinary differential equation whose solution is well known:

$$P(k_x, k_y, z) = A e^{ik_z z} + B e^{-ik_z z} \ . \tag{7.7}$$

Note that by expressing certain quantities in their related Fourier domains through repeated applications of the Fourier transform, we have moved from the wave equation, which is very difficult to solve, to a simple ODE.

Because of the way we have defined the coordinate system and the relation between time and space in the harmonic functions, the first term in

7.7 represents outward travelling waves (up from the floor) and the second term represents inward travelling waves (down toward the floor). However, the statement of the problem defined the half-space above the floor as being source free; no waves could be coming downward from above the floor. This boundary condition forces $B = 0$ in 7.7. *Application of this source free condition is crucial to the success of NAH.* Without it, we can't tell unambiguously whether a measured acoustic disturbance is an outgoing wave or an incoming wave.

To solve our equation, we need another boundary condition. The floor, or source surface, provides that condition. The pattern of waves on that surface is the only available source of acoustic energy in the fluid. The acoustic pressure right on that surface has some distribution over x and y that we describe via the spatial Fourier transform on those coordinates, that is

$$p(x, y; z = 0) = \frac{1}{4\pi^2} \int\int_{-\infty}^{\infty} P(k_x, k_y; z = 0) e^{ik_x x} e^{ik_y y} dk_x \, dk_y \ .$$

But $P(k_x, k_y; z = 0)$ is precisely the coefficient A of equation 7.7 (set $z = 0$ in this equation to see this). Thus, defining $P_0(k_x, k_y) = P(k_x, k_y; z = 0)$, we obtain an expression for the wavenumber domain pressure everywhere above the floor:

$$P(k_x, k_y; z) = P_0(k_x, k_y) e^{ik_z z} \tag{7.8}$$

Applying the inverse Fourier transform again, we obtain

$$p(x, y, z) = \frac{1}{4\pi^2} \int\int_{-\infty}^{\infty} P_0(k_x, k_y) e^{ik_x x} e^{ik_y y} e^{ik_z z} dk_x \, dk_y \ .$$

This equation immediately implies that the distribution of waves on the surface bounding the half-space *completely* determines the pressure throughout the half-space. This is the key to holography, which literally translates to "whole writing"; write or graph the solution on the boundary, and the whole field is determinable.

Through all of this explication, we haven't discussed the nature of the wavenumber k_z. If a plane, harmonic pressure wave is propagating into the half-space in some direction away from the floor, we describe its direction with the vector $\mathbf{k} = (k_x \mathbf{e}_x, k_y \mathbf{e}_y, k_z \mathbf{e}_z)$ where $(\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z)$ are the unit vectors in x, y and z. The size of $\mathbf{k}$ is determined in the usual Euclidean sense, $k^2 = k_x^2 + k_y^2 + k_z^2$. Since k is known for a given fluid and frequency, and k_x

and k_y are known from the distribution on the bounding surface, it is easy to see that k_z is definable in terms of them:

$$k_z^2 = k^2 - (k_x^2 + k_y^2)$$

Although k is fixed by the fluid and frequency, no such restrictions are placed on k_x or k_y; we can impose (or imagine) any wave distribution we desire on the bounding surface. This means that the quantity k_z^2 can be negative as well as positive, and that its root k_z can be either real or imaginary. Herein lies the key to *nearfield* holography.

Suppose that a harmonic wave of frequency ω exists on the floor with a wavelength much shorter than that supportable in the fluid at that frequency. It propagates across the floor with a *trace* wavenumber $k_t^2 = k_x^2 + k_y^2$. This trace wavenumber is greater than the fluid wavenumber k, so k_z becomes imaginary. Then the solution to equation 7.8 becomes

$$\begin{aligned} P(k_x, k_y, z) &= P_0(k_x, k_y)e^{i(ik_z)z} \\ &= P_0(k_x, k_y)e^{-k_z z} . \end{aligned}$$

This *evanescent* wave distribution on the floor decays exponentially in the z direction. This is the definition of an acoustical nearfield: wave distributions on a source cannot propagate to the farfield. Solutions to equation 7.8 must then be classified into one of two categories depending on the relationship of the trace wavenumber on the bounding surface and the fluid wavenumber at the same frequency:

$$\begin{aligned} P(k_x, k_y, z) &= P_0(k_x, k_y)e^{ik_z z} , \qquad k_t \leq k ; &(7.9)\\ &= P_0(k_x, k_y)e^{-k_z z} , \qquad k_t > k . &(7.10) \end{aligned}$$

The power of nearfield holography lies in the ability to describe the entire wave distribution on the source surface, including both evanescent and propagating contributions.

So how does all of this apply to measuring a hologram and making reconstructions? In equation 7.9, suppose the variable z was replaced by a specific value z_H, the distance to a hologram surface. Further, imagine that the pressure distribution was measured over all x and y at this surface, then Fourier transformed. By inverting equation 7.9 directly, the pressure distribution at the source surface $z = 0$ would be entirely characterized:

$$
\begin{aligned}
P_0(k_x, k_y) &= P(k_x, k_y, z_H)e^{-ik_z z_H} \qquad k_t \leq k \qquad (7.11)\\
&= P(k_x, k_y, z_H)e^{k_z z_H} \qquad k_t > k\ . \qquad (7.12)
\end{aligned}
$$

This is the informal definition of a reconstruction by back propagation. Note, however, that inversion is not a free lunch. Any such inverted system is an ill-posed problem, and questions of uniqueness of the solution are not trivial, nor can they be ignored. While we only touch on these issues in this document, it is easy enough to see what happens to P_0 in the case $k_t > k$ when z_H is large in relation to the wavelength λ_z. Since, in experiments, most of the measurement noise is manifested in the highest wavenumbers (where $k_t \gg k$), reconstructions can quickly become meaningless if z_H is chosen too large and appropriate wavenumber filtering is not applied.

7.2.2 A Formal Derivation of Holography

The following derivation is lifted mostly from [2]; the reader is encouraged to study that reference more closely.

We first give some definitions. We assume that some source is creating an acoustic pressure field $p(\mathbf{r}, t)$ in a space spanned by $\mathbf{r}$ over a time t which satisfies the homogeneous wave equation 7.1. We assume that some surface S bounds this space, and that there exists a Green's function $G(\mathbf{r}|\mathbf{r}_S)$ which satisfies the homogeneous Helmholtz equation 7.2 for $\mathbf{r}$ inside S and vanishing (or having a vanishing normal derivative) for $\mathbf{r} = \mathbf{r}_S$ on S. That part of S which is not at infinity is in practice a level surface of some separable coordinate system which is in close contact with the sources. In the sketch of Section 7.2.1 above, S was the "floor" of the semi-infinite half-space. Finally, we assume there is another surface H which is a level surface parallel to S for which either $p(\mathbf{r}_H, t)$ or its normal derivative can be measured. This is the hologram surface.

First, the data collected as $p(\mathbf{r}_H, t)$ are transformed to the frequency domain:

$$p(\mathbf{r}_H, \omega) = \int_{-\infty}^{\infty} p(\mathbf{r}_H, t)e^{i\omega t}\, dt, \qquad (7.13)$$

at which point the wave equation becomes the Helmholtz equation as shown previously. For a fluid in the space which will support acoustical waves with a compressional speed c, a wave propagating with frequency ω is defined as having a wavenumber $k = \omega/c$. In using the Helmholtz equation, we can deal with independent frequencies ω. We are now concerned with solving for $p(\mathbf{r}_S)$ from $p(\mathbf{r}_H)$ at a given frequency ω. Note that $p(\mathbf{r}_H)$ is a complex quantity, possessing both amplitude and phase (relative to a reference) information.

We begin the spatial processing by examining the Helmholtz Integral Equation. This equation derives from application of Green's Theorem to the Helmholtz equation, in this case for acoustic pressure. In its most general form, it has two terms:

$$p(\mathbf{r}) = \frac{1}{4\pi}\int_S (G(\mathbf{r}|\mathbf{r}_S)\frac{\partial p}{\partial n}(\mathbf{r}_S) - p(\mathbf{r}_S)\frac{\partial G}{\partial n}(\mathbf{r}|\mathbf{r}_S))\, dS \tag{7.14}$$

where $G(\mathbf{r}|\mathbf{r}_S)$ is the Green's function satisfying the Helmholtz equation which propagates waves from $\mathbf{r}_S$ to $\mathbf{r}$ in the space bounded by S, and $\partial/\partial n$ is the derivative in the direction normal to dS. For holography, we construct $G(\mathbf{r}|\mathbf{r}_S)$ as the sum of the free–space Green's function plus another term which will force G to zero on the boundary $\mathbf{r}_S$. This will then satisfy the homogeneous Dirichlet conditions on S, that is, $G(\mathbf{r}|\mathbf{r}_S) = 0$. This forces the first term in 7.14 to vanish, leaving only

$$p(\mathbf{r}) = -\frac{1}{4\pi}\int_S p(\mathbf{r}_S)\frac{\partial G}{\partial n}(\mathbf{r}|\mathbf{r}_S)\, dS\ . \tag{7.15}$$

Note that the integration is over the surface S, which we take to be the first two spatial coordinates described by $\mathbf{r}_S$. If S is a level surface of a separable coordinate system, and we fix that level by taking the third coordinate to be constant, then G can be found analytically and equation 7.15 becomes a two dimensional convolution integral.

In holography, we do not know the field on the surface S. We know it on another level surface H. Using generalized coordinates ξ_1, ξ_2, and ξ_3, we can express 7.15 as

$$p(\xi_1, \xi_2, \xi_3^H) = -\frac{1}{4\pi}\int_S p(\xi_1', \xi_2', \xi_3^S)\frac{\partial G}{\partial n}(\xi_1 - \xi_1', \xi_2 - \xi_2', \zeta)\, d\xi_1'\, d\xi_2'\ , \tag{7.16}$$

where the primed coordinates vary over S and the unprimed coordinates vary over H. In this expression, $\zeta = \xi_3^H - \xi_3^S$ is a constant (we know the distance from our holography surface to our source surface). In principle, this is a two dimensional convolution, as mentioned above. However, we know the pressure over H; how can we extract the pressure over S from inside the convolution?

The equation is invertible through the use of the *convolution theorem.* Without going into detail, the convolution theorem states that the product of the Fourier transforms of two functions is equal to the Fourier transform of the convolution of the functions themselves, that is

$$\int_{-\infty}^{\infty} \psi(x)\phi(x-y)dy = \mathcal{F}^{-1}[\Psi(\xi)\Phi(\xi)] \ ,$$

where $\mathcal{F}$ is the Fourier transform operator, and $\Psi(\xi) = \mathcal{F}\psi(x)$ and $\Phi(\xi) = \mathcal{F}\phi(x)$. If we take the two dimensional Fourier transform in ξ_1 and ξ_2 of equation 7.16, we get

$$P(\xi_3^H) = P(\xi_3^S)\,\widehat{G}'(\zeta) \tag{7.17}$$

where $\widehat{G}'$ is the Fourier transform of the Dirichlet Green's function relating the transformed surfaces by their separation distance, ζ. The other variables have been suppressed for clarity. We can now (informally) invert this expression and solve for the unknown $P(\xi_3^S)$ in terms of the known $P(\xi_3^H)$:

$$P(\xi_3^S) = [P(\xi_3^H)\,\widehat{G}'^{-1}(\zeta)] \tag{7.18}$$

This equation directly relates the source distribution to the distribution measured on H, but we don't yet know how to propagate the measurement to other surfaces $\xi_3 = \textit{constant}$. To do this, we rewrite equation 7.17 for the new surface ξ_3:

$$P(\xi_3) = P(\xi_3^S)\,\widehat{G}'(\xi_3 - \xi_3^S) \ . \tag{7.19}$$

We have an expression for $P(\xi_3^S)$ from equation 7.18, and we can substitute it into 7.19. Doing so, and taking the inverse Fourier transform over the first two coordinates, we arrive at the general expression for holographic reconstructions throughout the space bounded by S in terms of the measurement taken at H:

$$p(\xi_1,\xi_2,\xi_3) = \mathcal{F}^{-1}[P(\xi_3^H)\,\widehat{G}'^{-1}(\xi_3^H - \xi_3^S)\,\widehat{G}'(\xi_3 - \xi_3^S)] \ . \tag{7.20}$$

We have arrived at the fundamental expression for nearfield holography. It tells us what we need to do (theoretically, at least) to determine the distribution of waves on some separable surface. First, measure the waves on a surface parallel to the source surface (and close to it). Fourier transform the measured distribution over the surface. Develop the Fourier transform for the Dirichlet Green's function separating the hologram and source surfaces and the function separating the reconstruction and source surfaces. Invert the former and multiply both with the measured function for all wavenumbers. Inverse Fourier transform the result. Repeat this process at each individual frequency of interest. A set of reconstructions ensues.

We stress here that this inversion is informal. There is an entire body of mathematics literature which holds that formation of $\widehat{G}'^{-1}$ is at best risky, at worst impossible, and that the entire enterprise of reconstruction does not form a unique solution. We do not intend to delve into the details of "the Inverse Problem" here. Suffice it to say that the inversion works in an engineering sense, in that under the controlled circumstances of real NAH experiments and processing, reconstructions are similar to the "true" answer to within some error bounds which are tolerably small.

7.3 Holography in Cartesian Coordinates

We have, in essence, solved the problem of nearfield holography analytically. It remains to derive the Green's functions and their Fourier transforms for the coordinate systems of interest.

In Cartesian coordinates, the separable surfaces are planes. As in the informal sketch presented in Section 7.2.1, we align our axes so that the surfaces are planes in (x, y), separated by some distance $z_H - z_S$. The Green's function which satisfies the homogeneous Dirichlet boundary condition on z_S is given by

$$
\begin{aligned}
G(x, y, z|x', y', z') = \\
&\frac{exp[ik\sqrt{(x-x')^2 + (y-y')^2 + (z-z')^2}]}{\sqrt{(x-x')^2 + (y-y')^2 + (z-z')^2}} - \\
&\frac{exp[ik\sqrt{(x-x')^2 + (y-y')^2 + (z-z'-2z_S)^2}]}{\sqrt{(x-x')^2 + (y-y')^2 + (z-z'-2z_S)^2}} .
\end{aligned}
\qquad (7.21)
$$

(We do not develop this function here: it can be found in Morse and Feshbach

[5].) Fixing $z' = z_S$ and taking the normal derivative $(\partial/\partial z')$, we obtain

$$\frac{\partial G}{\partial n}(x, y, z|x', y', z_S) =$$
$$-2\frac{\partial}{\partial \alpha}\Big(\frac{exp[ik\sqrt{(x-x')^2+(y-y')^2+\alpha^2}]}{\sqrt{(x-x')^2+(y-y')^2+\alpha^2}}\Big)\Big|_{\alpha=\,(z-z_S)} \,. \qquad (7.22)$$

These expressions apply, of course, to the spatial domain. To apply them in our holography algorithm, we require the Fourier transform of $\partial G/\partial n$. Fortunately, it can be found analytically:

$$\begin{aligned} \widehat{G}'(k_x, k_y, z) &= e^{iz\,(k^2-k_x^2-k_y^2)^{1/2}} \,, \qquad k_x^2 + k_y^2 \leq k^2 \,, \\ &= e^{-z\,(k_x^2+k_y^2-k^2)^{1/2}} \,, \qquad k_x^2 + k_y^2 > k^2 \,. \end{aligned} \qquad (7.23)$$

The wavenumber quantities in the exponentials are the effective z wavenumbers. As in our informal sketch of holography, the issue of the trace wavenumber enters and gives rise to two exclusive solutions. As long as the trace wavenumber $k_t^2 = k_x^2 + k_y^2$ is less than the fluid wavenumber $k^2 = (\omega/c)^2$ at frequency ω, the waves distributed on surface S will propagate into the fluid. When the trace wavenumber exceeds the fluid wavenumber (the wavelength on the surface is less than that in the fluid), the wave on S decays exponentially along z into the fluid. In a two dimensional image of the wavenumber domain, the boundary formed by this relationship is the "radiation circle"; its radius is the trace wavenumber, k_t. This radius increases linearly with frequency (since the fluid wavenumber is proportional to frequency), forming a "cone" in wavenumber–frequency space.

Now that we are in possession of the proper expression for the Dirichlet Green's function, we can express the holographic reconstruction relation for planar coordinates. Substituting equation 7.23 into equation 7.20 yields

$$\begin{aligned} p(x, y, z) &= \mathcal{F}^{-1}[P(k_x, k_y, z_H)\, e^{ik_z\,(z-z_H)}] \,; \qquad k_x^2 + k_y^2 \leq k^2 \,, \\ &= \mathcal{F}^{-1}[P(k_x, k_y, z_H)\, e^{-k_z\,(z-z_H)}] \,; \qquad k_x^2 + k_y^2 > k^2 \,. \end{aligned} \qquad (7.24)$$

Note that z_S does not explicitly appear in this expression. This is due to the simplicity of the product $\widehat{G}'^{-1}(z_H - z_S)\,\widehat{G}'(z - z_S)$ in Cartesian coordinates; the term with z_S factors out. More generally, the surface S only establishes the range of validity of the final expression. While it could certainly be the z to which we reconstruct the data (and usually is), as the source plane, it does not play an explicit role in propagating waves from z_H to z.

7.4 Holography in Cylindrical Coordinates

Cylindrical coordinates are described in terms of axial distance, z, a circumferential angle ϕ about the origin axis and a radius from that axis of r. A bounding surface S consists of an infinitely long, right circular cylinder centered on the z–axis of radius r_S; only the space $r > r_S$ is describable by holographic reconstructions. A hologram measurement surface H would be over another right circular cylinder concentric to S at radius r_H.

In this coordinate system, the wavenumber domain consists of k_z, the axial wavenumber, n, the circumferential order, and $k_r = \sqrt{k^2 - k_z^2}$, the radial wavenumber. Note that k_r is analogous to the k_z of Cartesian coordinates in that it is derived from the fluid wavenumber and a trace wavenumber. This implies that it, too, can be either real and propagating (when $k_z \leq k$) or imaginary and evanescent (when $k_z > k$), as we will see momentarily in the solution. Note, too, that k_r does *not* depend on the circumferential order n. This does not imply that circumferentially oriented waves cannot be evanescent*like*; we will be able to say more about this when we examine the solution. Finally, note that while k_z and k are continuous variables (and k_r by extension), n is drawn only from the set of integers. To be continuous, a wave solution around the circumference of a circle must have an integer number of wavelengths. (Zero is a possible solution.) This has very strong implications for the use of discrete transforms in the algorithm for nearfield holography, as will be discussed in Chapter 8.

We take a slightly different route in developing the expressions for cylindrical coordinates than that employed for Cartesian coordinates. (This development is lifted from both [2] and [3].) Fixing the radius at r_H, we expand the pressure distribution there in terms of (orthogonal) cylindrical wavefunctions:

$$p(z, \phi; r_H) = \frac{1}{2\pi} \sum_{n=-\infty}^{\infty} \left(\int_{-\infty}^{\infty} P_n(k_z; r_H)\, e^{ik_z z}\, dk_z \right) e^{in\phi} \,. \tag{7.25}$$

The wavefunction coefficients $P_n(k_z; r_H)$ are found by taking the two dimensional Fourier transform of equation 7.25 and invoking the orthogonality conditions:

$$P_n(k_z; r_H) = \frac{1}{2\pi} \int_0^{2\pi} e^{in\phi} \left(\int_{-\infty}^{\infty} p(z, \phi; r_H)\, e^{ik_z z}\, dk_z \right) d\phi \,. \tag{7.26}$$

Because $p(z, \phi; r)$ was expanded in orthogonal wavefunctions, equations 7.25 and 7.26 can be seen to be a Fourier transform pair (noting that the inverse transform on n, which is discrete, becomes a Fourier series). In other words,

$$\begin{aligned} p(z, \phi; r_H) &= \mathcal{F}^{-1}[P_n(k_z; r_H)] \qquad \text{and} \\ P_n(k_z; r_H) &= \mathcal{F}[p(z, \phi; r_H)] \,. \end{aligned} \tag{7.27}$$

We now have expressions relating pressure wave distributions to the coefficients of cylindrical wavefunctions via the Fourier transform.

We now need to find how to propagate a known distribution radially from one surface to another. In equation 7.25, we fixed r_H and expanded in wavefunctions only over a cylinder of radius r_H. The general solution in cylindrical coordinates contains radial functions as well which are expressed in terms of the Hankel functions, $H_n^{(1,2)}(x) = J_n(x) \pm iY_n(x)$, where $J_n(x)$ and $Y_n(x)$ are the Bessel functions of order n and argument x of the first and second kinds, respectively. Since the holographic space is entirely exterior to the bounding surface S and is source free, only outgoing waves are allowed. This restricts the solution to $H_n^{(1)}(x)$, the Hankel function of the first kind of order n and argument x, which we will refer to here as simply $H_n(x)$. Equation 7.25 generalizes to

$$p(z, \phi, r) = \frac{1}{2\pi} \sum_{n=-\infty}^{\infty} \left(\int_{-\infty}^{\infty} P_n(k_z) H_n(k_r r) \; e^{ik_z z} \, dk_z \right) e^{in\phi} \tag{7.28}$$

where, as mentioned above, $k_r^2 = k^2 - k_z^2$. When $k_z > k$, the argument to the Hankel function, $k_r r$, becomes imaginary, and the Hankel function becomes a *Modified* Hankel function $K_n(k_r r)$. Whereas the Hankel function behaves asymptotically as $exp(ik_r r)$ (with cylindrical spreading), the Modified Hankel function behaves as $exp(-k_r r)$; when the radial wavenumber becomes imaginary, the axial trace waves no longer propagate, but become evanescent, as in the Cartesian case.

The radial wavefunction in equation 7.28 forms the Green's function for this coordinate system. It does not participate in the Fourier transforms on z and ϕ. We can be rewrite 7.28 for any radius, including r_H:

$$p(z, \phi, r_H) = \frac{1}{2\pi} \sum_{n=-\infty}^{\infty} \left(\int_{-\infty}^{\infty} P_n(k_z) \, H_n(k_r r_H) \, e^{ik_z z} \, dk_z \right) e^{in\phi} \,.$$

Note that the P_n are not functions of r, but eigenvalues of the cylindrical wavefunction expansion. The radial (propagation) information is contained in the Green's function, $H_n(k_r r)$. In other words, the pair of relations 7.27 held only for a fixed value of r. To relate distributions at different radii, we must express 7.27 with the radial propagator included. Doing this at an arbitrary reconstruction radius r and at the specific hologram surface r_H yields

$$\begin{aligned} p(z,\phi,r) &= \mathcal{F}^{-1}[H_n(k_r r)P_n(k_z)] \qquad \text{and} \\ p(z,\phi,r_H) &= \mathcal{F}^{-1}[H_n(k_r r_H)P_n(k_z)] \;. \end{aligned}$$

Solving for the coefficients $P_n(k_z)$ in the second expression and substituting into the first expression, we arrive at the relation for holography in cylindrical coordinates:

$$\begin{aligned} p(z,\phi,r) &= \mathcal{F}^{-1}[H_n(k_r r)H_n^{-1}(k_r r_H)P_n(k_z)] \;; \qquad k_z \leq k \;, \\ &= \mathcal{F}^{-1}[K_n(k_r r)K_n^{-1}(k_r r_H)P_n(k_z)] \;; \qquad k_z > k \;. \end{aligned} \tag{7.29}$$

These equations are valid for any $r > r_S$, the smallest cylinder which will enclose all the sources. Compare this expression with that of equation 7.24 in the previous section: only the Green's function has changed.

As an aside, we consider the evanescent*like* behavior of circumferential waves smaller than the fluid wavelength at a given frequency, as mentioned above. The higher the order n for a circumferential wave, the smaller the wavelength on a given cylinder, since $\lambda_\phi = 2\pi r/n$. When n becomes large relative to the argument $k_r r$, the Hankel function can be asymptotically approximated by

$$H_n(x) \approx \frac{1}{\sqrt{2\pi n}}\left[\left(\frac{ex}{2n}\right)^n - 2i\left(\frac{ex}{2n}\right)^{-n}\right] \,, \qquad x = k_r r \;. \tag{7.30}$$

When $x/n < 1$, the second term dominates, and the Hankel function decays as the nth power of radius. At a radius $r_0 = \lambda n/2\pi$, the first term begins to dominate, and the power law decay gives way to the cylindrical spreading farfield proportional to $1/\sqrt{r}$. This is demonstrated clearly in Section I.A.1 of [3].

7.5 Other Field Quantities

So far, we have concentrated on reconstruction using the Dirichlet Green's functions. In acoustics, this translates to the pressure field. However, the power of nearfield holography is greatly extended by the introduction of reconstructed velocity. In fact, normal velocity on the surfaces of cylindrical shells is probably the most important working quantity for structural acoustics research.

In addition, pressure and velocity reconstructions can be combined to provide acoustic intensity. This quantity is the key to source and sink location on radiating structures as well as computation of power flow and total radiated power. In many cases, this last quantity is used as a metric for comparing different cases, especially simulations using coupled Finite/Infinite/Boundary Element codes.

We won't provide detailed derivations of reconstructed velocity; instead, we simply state the formulæ needed to compute them.

7.5.1 Reconstructed Normal Velocity

There are two paths which will lead us to a reconstruction formula for normal velocity: through Newton's law and through the Neumann boundary conditions. They are, of course, different views of the same problem. We approach the topic through Newton's second law.

In analogy with the statement $F = ma$, we relate the product of the time rate of change of the fluid velocity (acceleration) and the density (mass) to the gradient of the pressure (force):

$$\rho \frac{\partial \mathbf{v}}{\partial t} = -\nabla p \,, \tag{7.31}$$

where each component of $\partial \mathbf{v}/\partial t$ is related to the affiliated component of the gradient operator. Assuming harmonic time dependence, we can directly relate the velocity to the pressure:

$$\mathbf{v} = \frac{1}{i\omega\rho} \nabla p \,. \tag{7.32}$$

This simple expression gives us a formula for deriving the operator for transforming from measured pressure to reconstructed velocity. Since we will be operating in the wavenumber domain, we Fourier transform 7.32 and substitute the propagated pressure as shown in 7.19. This yields

$$V_\eta(\xi_1,\xi_2,\xi_3) = \frac{1}{i\omega\rho}(\mathcal{F}\nabla)_\eta\left[\widehat{G}'(\xi_3-\xi_3^S)\,P(\xi_3^S)\right] , \tag{7.33}$$

where η represents the individual vector components. For radiation, the component in the direction normal to the source surface (the propagation direction) is of greatest interest. Since P is not a *function* of the normal coordinate, the normal gradient does not operate on it, but only on the Green's function propagator. Of course, the work here involves computing the transform of the gradient operator, then applying it to the propagator. More details may be found in [2] and [3]; we simply state the results.

In Cartesian coordinates, the normal velocity propagator appears only slightly modified from the pressure propagator of equation 7.23:

$$\begin{aligned} \widehat{G}_z'(k_x,k_y,z-z_H) &= \left(\frac{k_\eta}{k}\right)e^{ik_z(z-z_H)} ; \qquad k_x^2+k_y^2 \le k^2 , \\ &= \left(\frac{k_\eta}{k}\right)e^{-k_z(z-z_H)} ; \qquad k_x^2+k_y^2 > k^2 . \end{aligned} \tag{7.34}$$

In cylindrical coordinates, the normal component is in the radial direction. In this case, the propagator of equation 7.29 becomes

$$\begin{aligned} \widehat{G}_r'(k_z,n,r-r_H) &= -k_r\left[H_n'(k_r r)H_n^{-1}(k_r r_H)\right] ; \qquad k_z \le k , \\ &= -k_r\left[K_n'(k_r r)K_n^{-1}(k_r r_H)\right] ; \qquad k_z > k , \end{aligned} \tag{7.35}$$

where the prime on the numerators indicates the derivative of the function with respect to argument. Since the derivatives of Bessel functions are easily determined from the Bessel functions themselves via a recursion relation, these propagators are no more difficult to compute than the pressure propagators.

7.5.2 Acoustic Intensity and Radiated Power

With the pressure and normal velocity distributions available over a surface, the power flow per unit area, or intensity, is computable. The fundamental formula is simply

$$\mathbf{I} = \frac{1}{2}\mathrm{Re}(p\mathbf{v}^*) , \tag{7.36}$$

where $\mathbf{v}^*$ denotes the complex conjugate of the velocity and Re represents the real part of the complex product. This quantity is particularly valuable in the nearfield, where it can map power flow. By definition, in the farfield all power flow is outward.

Once the intensity is computed everywhere over the bounding surface, the total radiated power can be computed for each frequency. By definition, the power is the area integral of the intensity:

$$\Pi = \frac{1}{2} \int_S Re(p\mathbf{v}^*)\, dS \; . \tag{7.37}$$

For the planar system, $dS = dx\, dy$; for a cylindrical system, $dS = r\, d\phi\, dz$. Note that Parseval's theorem guarantees that we could also compute total power from the integral over wavenumber of the modulus squared pressure coefficients in the wavenumber domain. Such an integral would be limited to the supersonic wavenumbers, since they are the only ones that radiate.

Chapter 8

Implementation of the NAH Algorithm

In Section 7.2.2, we developed the basic recipe for reconstructing nearfield acoustical holograms. We transform pressure time series taken at a mesh of locations describing a separable coordinate surface to the frequency domain, then rearrange the data into holograms at each frequency. To perform reconstructions, we Fourier transform the measured distribution over that surface to the wavenumber domain. With the Fourier transforms of the Dirichlet Green's functions for the hologram and reconstruction surfaces in hand, we multiply the wavenumber domain hologram data by the Green's functions point for point over all the wavenumbers. We then inverse Fourier transform the result back to the spatial domain. We repeat this process at each individual frequency of interest and develop a set of reconstructions.

To accomplish this set of operations on a computer, every aspect of the process must be discretized. Discretization is straightforward, but it presents a set of challenges about which the experienced holographer must be aware. The side effects of discretizing operations which are modelled continuously impact not only the processing of holographic data, but the design of experiments intended to collect the holograms in the first place. The reader is once again referred to [2] and [3], but the definitive discussion of algorithms and implementation is presented by Veronesi and Maynard [4].

8.1 Transition to a Finite, Discrete Domain: Assumptions

We begin with a discussion of the issues involved in discretization, followed by the actual formulæ involved. We then examine how discretization impacts the experiments and the data in nearfield holography.

The two most important issues in implementation of holography are the truncation of the infinite domain to a finite domain and the discretization of the continuum to some finite mesh. The development of the expressions for nearfield holography depended on the bounding surfaces being of infinite extent. In planar holography, this is the entire plane at some z_S; in cylindrical holography, this is the infinite cylinder at r_S. In reality, we truncate this domain to some aperture of length L. In cylindrical holography, this forms a right circular cylinder of extent $-L/2 < z < L/2$. In order for the infinite integrals of the holography algorithm to remain accurate, we must assume that the function representing the measured quantity $\psi(z, \phi, r)$ *vanishes* outside this domain. This might seem ridiculous in the free field outside a finite, radiating shell. However, two things come to our rescue. First, one usually takes holography data as close to the source as possible (more on this below). Since the acoustic pressure levels are typically quite high in the region close to the radiator, they tend to fall below the measurement dynamic range within a relatively short distance (axially) from the body. Second, one usually "windows" the data (axially), forcing a smooth taper to zero value at the ends. Together, these conditions suffice to meet the vanishing function criterion. To help enforce the first condition, the aperture L must significantly exceed the length of the body under study. At NRL, we usually require a 50% overscan (i.e., $1/4L$ past each end of the shell) for cylindrical experiments. Aperture length will be discussed below in more detail.

The discretization of the measured pressure field depends on two more assumptions in the spatial domain and a concomittant assumption in the wavenumber domain. First, we must assume that the continuous wave field can be replaced by a discrete, finite set of values which sufficiently represent the field. These values are assumed to be the average pressure over a patch spanning the discretization interval. Further, it is assumed that the continuous wavefield can be constructed from a set of point measurements, with one point per patch. This assumption strongly impacts the *sampling interval*;

the wave field cannot vary in a manner which is not captured by the discrete samples. This can be formalized with Shannon's Sampling Theorem, whose primary result is commonly known as the Nyquist criterion: the sampling interval must be less than 1/2 the wavelength of the shortest wave expected in the system. Otherwise, aliasing of higher wavenumbers into the sampled field will result. This leads to the third assumption, that a discrete, finite set of wavenumbers is sufficient to characterize the Fourier transform of the wavefield. This criterion impacts both the aperture length and the sample interval, and will be discussed below.

8.2 Discretization of Integral Operations

Based on the assumptions presented above, we are ready to examine the effects of discretizing the continuous operations outlined in Chapter 7. The first of these is the two dimensional convolution known as the Rayleigh integral, which allows the wavefield at one surface to be propagated to another surface using the appropriate Green's function. We assume that the acoustic pressure field has been measured at a set of points describing a mesh within the measurement aperture, and that the size of this mesh is $M \times N$. (In cylindrical holography, this would be M points along the axis and N points around the circumference.) Then the convolution integral becomes (see [4])

$$\psi(p, q; r) = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} p(m, n; r_H) G(p-m, q-n; r-r_H) . \qquad (8.1)$$

In this expression, p and q are the indices of the reconstructed point at radius r, m and n are the indices of the hologram data values at radius r_H, ψ is the reconstructed quantity and G is the Green's function propagator associated with ψ. Veronesi [4] goes to some length in describing methods of computing G; for our purposes, we simply sample it in the wavenumber domain as we now describe.

The power (and speed) of GENAH lies in performing the convolution of equation 8.1 in the Fourier domain, where it becomes an inner product. To do this, we use the Discrete Fourier Transform (DFT), implemented through the Fast Fourier Transform (FFT) algorithm. The DFT is formalized as follows:

$$P_n(k_m;r) = \frac{1}{2\pi} \sum_{q=0}^{N-1} e^{-i(2\pi qn/N)} \sum_{p=0}^{M-1} e^{-i(2\pi pm/M)} p(z_p, \phi_q; r) \, \Delta z \Delta \phi \ , \tag{8.2}$$

where

$$\begin{aligned} z_p &= p\Delta z - \frac{L}{2} \ , \\ \phi_q &= q\Delta\phi \ , \\ k_m &= m\Delta k_z = m\frac{2\pi}{L} \ , \\ \Delta z &= \frac{L}{M} \ , \\ \Delta\phi &= \frac{2\pi}{N} \ . \end{aligned}$$

Here, L is the length of the axial aperture, M is the number of points along that length, and N is the number of angles around the circumference of the measurement cylinder, as used above. Substituting for $(1/2\pi)\Delta z\Delta\phi$ in 8.2, we obtain

$$P_n(k_m;r) = \frac{L}{MN} \sum_{q=0}^{N-1} e^{-i(2\pi qn/N)} \sum_{p=0}^{M-1} e^{-i(2\pi pm/M)} p(z_p, \phi_q; r) \ . \tag{8.3}$$

We could conceivably compute the transform of G in the same way (again, refer to [4]). However, since analytical expressions for $\widehat{G}$ are already available to us (see Sections 7.3 and 7.4), we find it much more convenient to simply sample those functions over the wavenumber set. Described by Veronesi [4] as Method 5, it is shown to be quite accurate for backpropagation over small distances. It is much less accurate in forward propagation over distances of more than a few wavelengths than some of the other methods he describes, but these instances are considered to be of far less interest to us.

8.3 Mitigating Side Effects in Truncation and Discretization

We now return to some of the issues brought up in examining the assumptions needed to truncate and discretize the hologram surface in Section 8.1.

First, we investigate the length of the measurement aperture, L. (In planar holography, both x and y require the same analysis.) One important criterion for L has already been described. This is the need for the wave field to vanish at the ends of the truncated aperture in order to satisfy the infinite convolution integrals. As mentioned, this requires that the measurement aperture overscan the shell by some amount (often about 25% past each end) to insure that the pressure falls to the bottom of the dynamic range. In addition (and often because this criterion fails, especially at low frequencies), we usually apply a tapered window to the data axially to ensure the vanishing constraint. Windowing as implemented in GENAH is discussed in greater detail in Section 5.2.1.

Another criterion in choosing the size of L is interference from replicated image sources. By their nature, DFT's are periodic over a period $2\pi M/L$. This implies that the Fourier transform representation of the hologram aperture repeats axially, and by analogy, the hologram itself repeats every L. If the hologram has significant contributions to the wave field near the ends of the aperture, the *images* of this source will contribute during the propagation process, interfering with the true propagation. Thus, L must be large enough to isolate the source and move its virtual images substantially away from the real aperture. Another way of looking at this is to consider the relationship between L and Δk_z:

$$\Delta k_z = \frac{2\pi}{L} . \tag{8.4}$$

If the source fills the aperture and L is not large enough, Δk_z will be too coarse to accurately represent the wave distributions; the field values at individual $k_m = m\Delta k_z$ become contaminated by the interfering image sources and have the wrong amplitudes and phases. This effect can sometimes be mitigated by a combination of windowing and zero padding. In zero padding, the data are extended to the next power of 2 points with zeros, doubling the effective aperture and the distance to the nearest images. When windows are used to taper the wave field to zero at the edges to meet the padded aperture, effective control of truncation errors and replicated sources is gained. However, these signal processing "tricks" are not a panacea for good experimental design. The best way to guard against the effects of spilling over the aperture and having replicated sources is to take preliminary test scans of the source at the lowest and highest frequencies of interest and at frequencies where the source strength is large, and to analyze their impact at the ends of the aperture.

It is of greatest importance to note that these issues do not come into play for the circumferential direction. This dimension is, by definition, finite. The convolution integral on ϕ is limited to the range $(0, 2\pi)$. Thus, there are no truncation or replicated image side effects in the ϕ direction.

Also of concern in designing a scan and later processing it is the sample interval, Δz. This interval directly determines the maximum available wavenumber in the axial direction,

$$k_z(max) = \frac{2\pi}{\Delta z} \tag{8.5}$$

As stated above, this interval must be sufficient to capture the smallest wavelengths likely to occur in the system. (As an aside, these are almost always the axial flexural waves on a cylindrical shell at the highest frequency of interest.) This choice is bounded by the Nyquist interval, 1/2 the minimum wavelength. In practice, this upper bound should be given wide berth: $\frac{1}{3}\lambda_{min}$ to $\frac{1}{4}\lambda_{min}$ is not unreasonable. In our controlled laboratory setting, however, we have been able to approach very close to the Nyquist limit and still get decent reconstructions.

Although one would like to go as high in wavenumber as possible, it must be remembered that the highest wavenumber contributions, almost always evanescent (or, equivalently, subsonic), grow the most rapidly in back propagation. This results in the destruction of the reconstruction's dynamic range. The solution to this is judicious use of wavenumber filtering, as described in Section 5.2.4. Nonetheless, there is no need to push $k_{z(max)}$ so high that the filters cut off well below it for all frequencies. This equivalently small Δz would mean that the system was too oversampled, a waste of very expensive resources. The additional errors introduced at high k_z due to discretization are treated in great detail in [4]: they are not discussed here.

The same principles of sampling do apply to the circumferential direction. Although the "aperture" (2π) is fixed, the number of angles sampled around the circumference must be sufficient to capture the smallest wavelengths expected in the system.

Finally, one is not, unfortunately, free to choose L and Δz independently of one another. Because of the Radix-2 FFT's used in GENAH , M and N must each be a power of 2. In the case of $M = L/\Delta z$, the choice of either L or Δz constrains the other quantity to within a power of 2. This often calls for some compromise. Our priority is usually placed on Δz, since many of the effects of the truncated aperture can be mitigated by windowing and zero padding, as described above. Most of our scans have been either 64 or 128 points in length; 32 is seldom enough for meaningful shells and 256 is often too expensive in terms of acquisition time. We have almost always used 64 angles circumferentially due to the use of point drives in the shells; in cases of ring drives or axisymmetric scattering, 32 or 16 angles sometimes suffice.

8.4 Resolution, Dynamic Range and Numerical Precision

We have mentioned several times previously the need for and the positive effects of locating the hologram measurement surface as close as possible to the source surface. This section will quantify that separation distance ($d = z_H - z_S$ in Cartesian coordinates, $d = r_H - r_S$ in cylindrical coordinates). It turns out to be directly related to the achievable wavelength resolution and to the measurement dynamic range. Thus, the value of d plays a part in choosing Δz and L as well.

Assume that a holographic measurement system has a given dynamic range D, measured in decibels (i.e., $20log(P_{max}/P_{min})$). For a digital acquisition system, the theoretical best–case baseline is based on the number of bits in the A/D converter. Accounting for the fact that the acoustic pressure has both positive and negative values (thus giving up one bit to the sign), a 10-bit converter gives a precision of 1 part in 512 (about 54 dB), a 12-bit converter gives 1 part in 2048 (about 66 dB) and a 16-bit converter gives 1 part in 32768 (about 90 dB). Of course, transducer, preamplifier, amplifier and timing/positioning/phase noise all reduce this theoretical value, but they differ for any two installations. Hopefully, their contributions can be kept to less than 12 dB (the two least significant bits).

For a given evanescent wave component at wavenumber k_r', the field will decay exponentially as the transducer is moved away from the source surface. At some distance, the component's amplitude will fall below the minimum threshold of the measurement system. This is how dynamic range relates to physical distance:

$$10^{-D/20} < e^{-k_r' d} \,. \tag{8.6}$$

The achievable spatial resolution R_z is bounded by the Nyquist rule, as discussed above:

$$R_z \leq \frac{\lambda'_z}{2} = \frac{2\pi}{2k'_z} \ . \tag{8.7}$$

Assuming that this highest axial wavenumber is well above the acoustical wavenumber for the highest frequency (usually a good assumption in shells), this gives an upper bound for $k'_r = \sqrt{k'^2_z - k^2} \approx \pi / R_z$. Taking the common logarithm of both sides of 8.6 and substituting for k'_r, we obtain

$$R_z = 20\pi log_{10}(e)\Big(\frac{d}{D}\Big) \approx 27.3\Big(\frac{d}{D}\Big) \ . \tag{8.8}$$

This expression gives us the guidance we need to choose d based on our knowledge of D and the shortest likely wavelength on the structure, λ_{min}. As an example, assuming $D = 55$ dB (about right for a 12–bit system with good positioning), equation 8.8 yields $d \approx \lambda_{min}$. This could be very small for a thin shell at high frequencies—possibly just a few millimeters! This is where the process of balancing the hologram parameters for experimental design begins.

How does this impact GENAH ? For the most part, the quality of the data (its dynamic range) will influence both the maximum distance over which holograms may be back propagated and the magnitude of the cutoff wavenumber for the wavenumber filters. If the reconstruction distance is fixed (say, back to the source surface), then the impact is felt in the filtering and thus in the available resolution. If d is larger than the data quality can support, the GENAH user may find that the filters must be so restrictive that the reconstructable resolution and possibly the frequency range become more limited than desired. This is why it is so valuable to preview the data before reconstruction and to perform forward–transform–only analysis (Option 1) to examine the wavenumber content and its noise level.

Chapter 9

Analysis of FORTRAN Code in GENAH

In this chapter, we examine the FORTRAN code in GENAH and discuss exactly how it implements the holographic algorithm discussed in the previous chapters. For a list of subroutines and their arguments, see Appendix D.

9.1 Variable Declarations

GENAH uses the `implicit none` qualifier to force explicit declaration of all parameters. All of the subroutines provided by NRL follow this convention.

The first set of declarations is a group of parameters. The first four control array sizes: `nrkcbins`(250) is the number of wavenumber filters that can be read in; `maxn`(34) is the maximum order of Bessel function needed (it should be $n_{max}+2$, where n_{max} is the highest positive circumferential order allowed); `ldz`(256) is the leading dimension of the hologram data arrays (the maximum number of axial points allowed); and `sdz`(128) is the second dimension of the hologram data arrays (the maximum number of circumferential points allowed). The other three parameters are: `twopi`, simply the value of 2π; `rtd`, a radians–to–degrees conversion factor; and `rhoc`, the specific impedance of water in metric units.

Following the parameter declarations are the declarations for all of the internal variables used by GENAH grouped in order of size; Integer, Real, Complex, Real*8 and Complex*16. Within these groups, variables are declared roughly in alphabetical order, but also grouped according to usage in the program. The data arrays are all complex. Double precision is used only in computing the Hankel functions—for large orders and arguments, the Bessel functions grow quite large, but their ratio (needed for the propagator, see Section 7.4) remains of small order.

The single `INCLUDE` file for GENAH is the set of declarations and common blocks for the data file header. They are numerous and confusing. For clarity's sake, they were collected into their own file. For more information on the Header and its contents, see Appendix B.

The variables used in the Radix-2 Fast Fourier Transform (FFT) in GENAH are gathered separately from the general declarations for ease of manipulation. Note that while the arrays are of length `ldz`, the indexing vector `i2` is of length log_2(`ldz`). Since the compiler cannot perform the logarithmic operation, this value must be altered by hand if `ldz` is changed.

Finally, the two functions `cang` and `windo` are declared external to satisfy the explicit declaration criterion.

9.2 Input Data File and Parameters

The next section of GENAH concerns opening the input data file (pressure holograms) and setting parameters, many of which are flags which control decisions in the processing operations. The details of the inputs expected from the user are not covered here; they are discussed at length in Appendix C. However, it is appropriate to discuss some of the variables and their settings here.

There are a set of counters which describe the extent of the hologram mesh to the various `DO`–loops throughout the code. In particular, `naxpts`, the number of points in the axial direction (also the leading dimension of the data arrays), has three versions. This is because, as a result of the zero padding option, the input, working and output arrays may all have different lengths. The second dimension of the array, `ncirpts` does not vary in this way.

The choice of axial spatial windowing determines how the array `tuk` is loaded. "No window" actually creates the so–called *boxcar* function, with amplitude 1.0 over the aperture. The Tukey, Hanning and Hamming windows are discussed in more detail in Section 5.2.1.

Depending on the choice for zero padding, the number of axial points to be used in processing is set. Given that and the physical lattice spacing `alat`, the wavenumber increment `delk` is set according to the relation $\Delta k_z = 2\pi/(N_z a_z)$.

The frequency range and increment available in the hologram data file are announced. Based on these, the user can request which holograms should be processed. The resulting inputs become the frequency loop parameters.

The next section of code deals with setting the wavenumber domain filter parameters, discussed in Appendix E. If the option for a frequency–dependent filter file is chosen but the number of filters in the file exceeds the array limits set by parameter `nrkcbins`, a fatal error is flagged and the program is terminated. Also, circumferential order "low–pass" filtering can be set here.

The next section loads the all–important flag `ifqout`, the process control flag. The user will note that it is used throughout the remainder of the GENAH code to decide where the processing path should go next. While the current implementation of this decision tree is archaic, it has proven to be robust. It may be altered in a future release. Based on the values of `ifqout` and the zero padding flag `izadd`, the output leading dimension counter `naxpts_out` is set.

Next, the physical parameters which describe the hologram and the reconstruction surface are listed, and the user can alter them if necessary, for example if the reconstruction surface is to be at some radius other than the source surface. Again, more details are available in Appendix C.

When the user gives the name for the output data file(s), the code formats the names with the appropriate extension. It then opens the files for writing. Note that the open status is set to `'unknown'`: if the file exists, it will be overwritten; if it does not exist, it will be created. The header block is then loaded with the input hologram file header data after modifying it appropriately. To avoid conflict with common block quantities (such as `nsteps` and `fstart`), temporary variables are used to hold the input file values until the output file headers have been written.

9.3 The Main Loop

This is the heart of the GENAH program; the loop over the individual frequency bins for the processing of each hologram. The loop (and thus the current frequency) is controlled by the parameters `ihstart` (the index of the first hologram of those available to be processed), `ihno` (the total number of holograms to be processed) and `ihstep` (the step interval, or the increment to the hologram index variable `ihol`). Note that the top hologram is *not* `ihno` itself, but `(ihno - 1) * ihstep + ihstart`.

9.3.1 Filter Parameters

If the wavenumber filter is constant (independent of frequency), it is set. Otherwhise, the available list of filter parameters is searched for those which apply to the current hologram. This is done by finding the two bin pointers in the filter file which bracket the current hologram index. After the hologram index reaches the top bin pointer, it is no longer reset; the filter parameters remain set at the values specified for the top bin pointer.

9.3.2 Current Frequency and Wavelength

The current hologram index is used to compute the center frequency of the hologram frequency bin, based on the frequency of the first hologram in the input file and the bin width of that file as `freq = delf_in * (ihol - 1) + fstart_in`. The wavelength of an acoustic wave in water at that frequency is then computed using a sound speed of 1480.82 m/s (clean water at atmospheric pressure at 20 degrees Celsius). If the data from the input file is taken under other conditions, the value of the sound speed should be altered. It is conceivable that an open slot in the header (see Appendix B could be used to store the temperature and depth of water at the time the data was taken. Since these are closely controlled in NRL's laboratory environment, they have been held constant.

9.3.3 Reading in the Hologram

At this point, the data for the hologram pointed to by index `ihol` is loaded. The outermost `IF` block allows for the number of axial points to be any power of 2 up to and including 256. If other values are needed, this section must be modified. The first step within each block is to call the subroutine `filread_ieee`.

For any number of axial points M less than 256 (but still a power of 2), a temporary array `zdummy` is filled with the input data. If the zero padding flag `izadd` is set, the data array `zdat` is padded with $M/2$ zeros at the beginning and end of each scan line, and the data, multiplied by the windowing vector `tuk`, is loaded into the center M values. Otherwise, the data is windowed and loaded directly into the first M locations (for each column) of `zdat`.

The case of 256 axial points is a bit different. Since the data arrays are limited to 256 values in the leading dimension, scan lines of this length cannot be zero padded. Also, the temporary array `zdummy` is not used. Data are loaded directly into `zdat` and then windowed with vector `tuk`.

9.3.4 Conjugation, Maximum Search and Wavenumber

With the data loaded and windowed, the hologram is searched for the data point with maximum modulus, the index of its location in the mesh, and the complex phase of that value. This information is announced to the user's screen and copied into the ASCII output information file for future reference.

If the data need to be conjugated (see Section C), it is done in this loop. Use of the `IF` test on the conjugation flag `iconj` *inside* the loop is inefficient, but since this loop is executed only once per hologram, it does not matter very much.

Before the hologram processing begins, the current fluid wavenumber is computed, as well as some constants based on it that come into play in the propagation section. Also, the source and hologram radii are reset here.

9.3.5 Forward FFT, Axial Direction

The data must now be forward transformed to the wavenumber domain. To reduce the time spent in the transform process to a minimum, all of the Fourier weights (cosine and sine factors) and the butterfly bit reversal index are precomputed in routine `wtrev`. The "direction switch" `isw` is used to mark the sense of the complex coefficients and thus whether it is a forward or backward transform. In this case, setting `isw` to 0 indicates that it is a forward transform.

NOTE: The sense of the forward transform for spatial data is held to be the *opposite* of that for temporal data. This is a critical issue. An outward travelling wave is described by the function $exp[-i(\omega t - kr)]$; the signs of the transform kernels are opposite for temporal $e^{-i\omega t}$ and spatial e^{ikr} data. This *must* be taken into account in preprocessing data for GENAH !

The forward transforms (and all subsequent transforms) are performed on all the columns and then all the rows, taking advantage of the linearity of the Fourier transform operator in computing the two dimensional transform. The axial–coordinate data corresponds to the columns of the data array, and they are transformed first. Since the FFT routine employed here is in–place (destroying the input vector and replacing it with the result), the columns (rows) are loaded one at a time into a transform vector `zf`. The vector is transformed and loaded back into a result array.

In loading the result array, the vector is rotated by half the aperture, and all the even indices are multiplied by -1. The first process rotates the wavenumber domain so that the zero wavenumber is in the center of the array rather than in the corner. The second process, which is equivalent to multiplication by the Nyquist wavenumber, is then needed to properly phase all the contributions for plotting purposes. Otherwise, the signs of the components toggle positive and negative. This is self–correcting when a forward transform is followed by an inverse transform, but when processing (such as Options 1, 2 and 4) stops after the forward transforms, the toggling must be removed to properly visualize the transforms. If the user were to *always* do full reconstructions (back to the spatial domains), these loops could be removed.

9.3.6 Forward FFT, Circumferential Direction

If Processing Option 4 is in effect, no forward transforms are performed in the circumferential direction. The code jumps to markers 79, 210 and 215, at which point the $P(k_z, \phi; r_H)$ data are written to the output file. Likewise, if only a single scan line is available (`ncirpts` = 1), no circumferential transforms are performed. The code jumps to marker 250 to begin propagation (assumed to be for order $n = 0$ only).

Otherwise, the forward transforms are performed on data in the circumferential direction. Each row of the data array is loaded into the transform vector **zf**, transformed in place, and returned to the data array after having been rotated and toggled. (See Section 9.3.5 for an explanation of this process.)

When the forward transforms are complete, the weighting and bit reversal vectors are reset for inverse transforms in the axial direction, using **isw** = 1. This section of code should really precede the backward transforms locally; it is located at this point for historical reasons.

9.3.7 Computation of the Propagator

If Processing Options 1 or 4 are in effect, no propagations are applied to the data. The code jumps to markers 210 and 215, at which point the $P_n(k_z; r_H)$ or the $P(k_z, \phi; r_H)$ data are written to the output file.

Otherwise, this is the section in which the propagators for the current hologram frequency are generated and their dot product with the wavenumber domain data is formed. Propagators are computed as vectors whose indices are the circumferential orders n. The outer loop is over axial wavenumber index. The axial wavenumber is computed using this index and the interval wavenumber **delk**. The square of the radial wavenumber, called **tst** here, is computed. It serves as the switch for propagating or evanescent waves (see Section 7.4).

The first block executes if the radial wavenumber is real (propagating waves). In that case, the (**VAXMATH**) Bessel functions **dbesy** and **dbesj** are called to load vectors **by**, **by0**, **bj** and **bj0** at the hologram and source radii, respectively. The functions are computed up to order **maxn**, which is a parameter set to $|n_{max}| + 2$ (see Section 9.1). The Hankel function vectors are then formed according to $H_n(k_r r) = J_n(k_r r) + iY_n(k_r r)$ and stored in **bh** and **bh0**.

An inner loop over circumferential orders is executed next. A pointer **nn** into the Hankel function vectors is created to translate between the loop counter and the order comprising that row of the data array. (Recall that the propagators are symmetrical in wavenumber about the zero wavenumber.) The circumferential wavenumber is computed for the sake of the wavenumber filter. An offset is added to the Hankel pointer since **FORTRAN** arrays are numbered from 1, not 0. The pressure propagator **grnp** is computed as the product of the ratio of Hankel functions (see Section 7.4) and the wavenumber filter **windo** for the current axial wavenumber and circumferential order (see Section E). The propagated pressure value for the current circumferential order and axial wavenumber is then computed.

The velocity propagator **grnv** is computed in a similar manner, except that the formula is slightly different for $n = 0$ than for $n \neq 0$. This is due to the fact that the numerator of the propagator is the first derivative of the Hankel function. This is most easily computed using the recursion relation $H_n'(z) = (1/2)(H_{n-1}(z) - H_{n+1}(z))$ for $n > 0$ and $H_0'(z) = -H_1(z)$. The reader will also note the presence of the constant $ik_r/2\rho ck$, satisfying Newton's law for conversion from pressure to velocity (see Section 7.2.2). Again, the function **windo** is called to compute the value of wavenumber filter for the current wavenumbers. The reconstructed velocity value is then computed.

Because the propagator is symmetric, the outer loop counts over only half the axial wavenumbers. In this inner loop, the symmetric axial wavenumber component is also computed using the current values of the propagators.

The second block of this section is used if the radial wavenumber is imaginary (**rk** < 0) and the components are evanescent. In this case, calls are made to **dbesk** to compute the Bessel functions of the Third Kind. Otherwise, the block proceeds identically to that for the propagating waves as just described.

9.3.8 Backward FFT, Axial Direction

If Processing Option 2 is in effect, no inverse transforms are performed. The code jumps to marker 215, at which point the $V(k_z, n; r_S)$ are written to the output file.

Otherwise, the inverse transforms are begun on data, first in the axial direction. At this point, both pressure (**zp**) and velocity (**zv**) data must be transformed, so there are two sets of transforms within the loop on circumferential order. Within that loop, if Processing Option 5 is in effect, no inverse transforms are necessary on pressure, since only velocity is output. The code jumps to marker 230, at which point the inverse transforms on velocity proceed. The transforms themselves are identical in structure to the forward transforms, discussed above, with the exception that the direction switch **isw** is set to 1 for inverse transforms.

9.3.9 Backward FFT, Circumferential Direction

Following the inverse transforms in the axial direction, the code jumps to marker 215 for Processing Option 3, at which point the $V(z, n; r_S)$ data are written to the output file. Likewise, if only one scan line is being processed (synthetic axisymmetry), the code jumps to marker 250. At this point intensity and power are computed, and pressure, velocity and intensity are written to their respective output files.

The weighting and bit–reversal vectors are next loaded for the inverse transform on circumferential orders. As in the axial inverse transform, both pressure and velocity must be treated, so there are two sets of transforms. If Processing Option 5 is in effect, no inverse transforms are necessary in the circumferential direction. The code jumps to marker 240, at which point the individual orders are written to the output file. This is discussed in more detail in Section 9.3.10 below.

It is not immediately obvious, but only Processing Option 0 (full reconstruction) has survived to the point of inverse transforms on circumferential order. The transforms proceed exactly as they had in the forward transforms on ϕ, except that the direction switch `isw` is set to 1.

When the transforms are complete, the code jumps to marker 250, at which point the intensity and power are computed and pressure, velocity and intensity are written to their respective output files. This is described in detail in Section 9.3.11 below.

9.3.10 Limited Order Velocity Reconstruction

Processing Option 5 differs markedly from the other options. Instead of writing a single reconstruction for each frequency, this option writes $\mathtt{nmax} - \mathtt{nmin} + 1$ velocity reconstructions for each frequency. However, the reconstruction is not a function of circumferential order itself, that is, $V(z, n; r_S)$ as in Option 3. Instead, for each order `nn` between `nmin` and `nmax`, both the positive and negative orders are loaded by themselves into an otherwise empty vector and inverse transformed. This yields the reconstructed velocity $V_n(z, \phi'; r_S)$, where the ϕ' is used to indicate that only orders $\pm n$ are present in the velocity field.

Again, it is emphasized that for each frequency bin, Processing Option 5 writes $\mathtt{nmax} - \mathtt{nmin} + 1$ entries into the output file. Thus, output files will be much larger in size than would otherwise be the case. More importantly, when plotting or post–processing, the user should remember that only the first reconstruction will have the correct frequency associated with it. The number of orders for each bin would have to be used to compute its frequency, which is not a standard procedure in most post–processing programs.

When the inverse transforms are completed for each order of interest, the reconstructions are written to the output file. The code which does this is discussed in more detail in Section 9.3.12 below.

9.3.11 Intensity and Radiated Power

The acoustic intensity is computed exactly as prescribed in equation 7.36 of Section 7.5.2. At each lattice location `(i,j)`, the value of the pressure `zp(i,j)` is multiplied by the *conjugate* of the value of the radial velocity `zv(i,j)` and reduced by half. The complex result is stored in array `zdat`.

Prior to the double loop over the aperture, the accumulator `power` and a maximum intensity store `pmax` are zeroed. Within the loops, the real part of the intensity at each lattice point is extracted from `zdat(i, j)` and stored in `pterm`. This term is compared against `pmax`; if it exceeds `pmax`, the value of `pmax` is reset to `pterm`, the current (now maximum) intensity. The value of `pterm` is then added to the `power` accumulator. Note that this is a crude rectangular quadrature approximation to the integral of equation 7.37. However, as discussed in Chapter 8, the mesh should be fine enough to adequately represent the wave structure of the field. This implies sampling sufficient to allow the rectangular quadrature to approximate the power integral. After the loops are complete, the power accumulator is multiplied by Δz (`alat`) and $\Delta\phi = 2\pi r/n_\phi$ (`radius` and `ncirpts`). These represent the differentials dz and $d\phi$ in the integral of equation 7.37. Note that both `alat` and `radius` are multiplied by `conversion_factor`: this guarantees that the power is in SI units regardless of the units of the distance measures.

The maximum intensity and total power in the hologram are reported to the ASCII output file if Processing Option 0 (Normal Reconstruction) is in effect. Next, the intensities throughout the hologram are searched for any whose real part is within a fraction of the maximum. This fraction is preset in the code as parameter `cutof`. In the current version, `cutof` is set to 0.7 or 70% of the maximum real part of the intensity. Every point in the hologram whose intensity exceeds `cutof` has its pressure (magnitude and phase), velocity (magnitude and phase), intensity (magnitude) and the difference between the pressure and velocity phases written to the output file along with the indices of its location in the mesh. The phase difference is useful in determining how reactive the field is; as the difference approaches $\pm 90°$), the intensity is becoming purely reactive and no power is radiating. These values give an indication of how energy is distributed throughout the wave field. If only a few points are listed, then the maximum point (usually the drive point) stands alone. If many points are listed, then the source is probably near some sort of resonance where most of the surface is strongly contributing to the field. In essence, the quantities written to the ASCII output file give a qualitative feel for the character of the wave field.

9.3.12 Writing the Reconstructions: Spatial Domain

With all computations completed for Processing Option 0, the reconstructions can now be written to disk. Holograms are written using subroutine `filwrite_ieee`. They are written in fixed record length binary format (see Appendix A for details). The record lengths for all holography files are fixed at 128 longwords (512 bytes), so different numbers of records must be used for different numbers of axial points. A 128–longword record will hold 64 complex values. For cases where the number of axial points per scan line is less than 64 (but still a power of 2), more than one scan line can be written to a record, so the number of records needed to write an entire hologram is computed as

$$\texttt{num\_recs} = \texttt{ncirpts}/(64/\texttt{naxpts\_out})\ , \qquad (9.1)$$

where integer arithmetic is enforced. For the cases where the number of axial points per scan line is 64 or 128, the number of records is simply the number of scan lines, `ncirpts`. In that case, the subroutine uses the header value `nsteps(1)`, stored in the `COMMON` block, to determine how many records are needed.

If the input data have been zero–padded, there is no need to write out the padded region of the reconstructions. In that case, only the interior values are written out. This is accomplished by loading the values into the array `zdummy` and writing that array rather than the data array. In either case, three arrays are written: pressure to logical unit 8, velocity to logical unit 2 and intensity to logical unit 3. Once the data for the current reconstructions are written, the code jumps to marker 220, after which the main loop over frequency bin is terminated. If the current frequency was the last to be processed, all files are closed and GENAH terminates.

9.3.13 Writing the Reconstructions: Wavenumber Domain

Marker 215 is the jump point for output for Processing Options 1, 2, 3 or 4 (wavenumber domain output). As with spatial domain data, holograms are written using subroutine `filwrite_ieee`. The same 128–longword records are used for this data. Output for wavenumber domain data is the same as that for spatial domain data with one exception. The case of 256 axial points is allowed, since zero–padded data has interpolated values throughout the entire wavenumber domain. In other words, no distinction is made as to whether the data have been padded or not in the wavenumber domain. Processing Options 1, 2 and 4 fall into this category. In each case, the data are written to logical unit 2.

The case for Processing Option 3 ($V(z, n)$) is similar to that for spatial domain data in that if the data are zero–padded, only the center values are written. As such, it has its own processing block. In this case, the velocity data are written to logical unit 2.

Once the data for the current reconstruction are written, the code passes marker 220, after which the main loop over frequency bin is terminated. If the current frequency was the last to be processed, all files are closed and GENAH terminates.

Part III

Appendices

Appendix A

The Holography File Format

The input and output files for GENAH are described generally in Section 6. This appendix formalizes that discussion in sufficient detail to allow the user to create or read files in that format. If the user wishes to use some other scheme, the GENAH code can be altered. For input, the subroutine `filread_ieee` would need to be replaced. For output, the subroutine `filread_write` would need to be replaced.

All GENAH data are stored in an unformatted, binary fashion. They are not transportable across different types of computers (such as Sun, VAX, PC and so on) or operating systems (such as Unix, VMS, DOS, and so on). The data are written and read using the `FORTRAN` direct access method with a record size of 128 longwords (512 bytes). Note that a program cannot read the data using records whose length is an integer multiple of 128 longwords due to the presence of the header, which offsets the record pointer. The data can be read and written using the file manipulation routines contained in the `stdio` library of most C compilers.

Holography data can be described in either a deconstructive or a constructive way. From the top down, a file is made up of a header and all the holograms. A hologram is made up of scan lines. A scan line is made up of step locations or "points". A point contains 1 complex datum representing the calibrated value of pressure (velocity, etc.) per unit input. From the bottom up, a scan line is made up of one or more records. A hologram is made up of the records for all the scan lines at one frequency. A file is made up of the records for the holograms for a set of frequencies and the header.

A GENAH record is 512 bytes long. This is declared in the `FORTRAN-77`

OPEN statement as `recordsize=128` longwords at 4 bytes per longword. This record can represent 64 complex numbers at 2 longwords per number. Both input and output files are stored in direct access mode with this fixed record length. To access a record, a pointer (called an `associatevariable` in FORTRAN-77) must be declared in the OPEN statement. The OPEN statement should also include the keywords `form='unformatted', access='direct'`. An example is shown in the code fragment at the end of this appendix.

For scan lines of $M < 64$ steps (but still a power of 2), N scan lines are packed into each record according to the rule $N = 64/M$. For scan lines of $M = 64$ steps, each scan line constitutes a single data record. For scan lines of $M > 64$ steps (but still a power of 2), N records are used for each scan line where $N = M/64$. So, for instance, two 32–step scans would fit into a single record, a 128–step scan would require two records and a 256–step scan would require four records. A 154–step scan would require three records, although the third record would be padded out with 38 empty entries. The number of records in each hologram is equal to the number of scan lines times the number of records per scan line. A typical cylindrical hologram of 128 points per scan line and 64 scan lines would require 128 records of 512 bytes each, for a total of 64 kilobytes of storage.

Each file written in NAH format has a 512-byte header. Record number 1 of each file (whether input or output) is dedicated to this header record. The contents of the header are described in detail in Appendix B.

The size of a file in bytes is exactly computable. It is equal to the number of holograms (frequency bins) times the number of scan lines per hologram times the number of records per scan line times 512 bytes per record plus 512 bytes for the header.

The following code fragment demonstrates the key elements in writing data to a file in GENAH format. Reading data is exactly the same, where `write` statements are replaced by `read` statements (see the GENAH subroutine `filread_ieee.f`.

The code is self-explanatory with the exception of variables `i1` and `i2`. It is up to the user (code) to decide which values along the first coordinate of the hologram should be written at each loop. For example, suppose a cylindrical hologram has 128 steps per scan line (two records for each line) and all values should be written. The loop on `k` will execute twice. On the first pass, `first_x(1)` should generate the value "1" and `last_x(1)` should generate the value "64". On the second pass, `first_x(2)` should generate the value "65" and `last_x(2)` should generate the value "128". Suppose, however, that the internal representation of the hologram has been padded out to 256 steps per scan line, but only the center 128 values occur over the original hologram aperture. Only two records are required for the 128 points of interest, but `first_x(k)` should generate the values "65" and "129", respectively, and `last_x(k)` should generate the values "128" and "192", respectively. Then the control values `i1, i2` will properly point into the hologram array in the `write` statement's implied `DO` loop. Note that the current version of GENAH applies a brute force method of accomplishing this based on the processing path. A more flexible method will be used in an upcoming release. Note, too, that this discussion only applies to *writing* a hologram; when reading, the hologram array is just loaded beginning at the first value along scan lines.

```
c
c  -- Open the file as direct access, unformatted, 512-byte records.
c
      open (unit=iunit, file=filename, status='unknown',
     &      form='unformatted', access='direct', recordsize=128,
     &      associatevariable=nrec)
          .
          . (Load parameter values into 512-byte header array)
          .
      write (iunit, rec=1) ihead    !Write the header into record #1
c
      do hol_count = 1, nholograms  !Loop over frequency bins
          .
          . (Process holograms)
          .
         do j = 1, nscan_lines      !Loop over scan lines
            do k = 1, recs_per_scan !Loop over multiple records/scan line
               i1 = first_x(k)      !User must decide which points get
               i2 = last_x(k)       ! written from a scan line.
c
c  -- Set the record number pointer.
c
               nrec = ((hol_count - 1) * recs_per_scan * npts_y)
```

```
     &                 + recs_per_scan * j
     &                 + (k - 1)
c
c  -- Write the record using Implied DO along the 1st coordinate.
c
               write (iunit, rec=nrec) (zhol(i, j), i = i1, i2)
            end do                    !End loop over records/scan line
         end do                       !End loop over scan lines
      end do                          !End loop over frequency bins
```

Appendix B

The Holography File Header

B.1 Introduction

This section describes in detail the structure and contents of the header block prepended to any GENAH data file. The block is one record long, for a total of 512 bytes. (See Appendix A for more information about records.)

The header was originally developed for use on a PDP-11 series computer whose native word length was 16 bits, not 32 bits as in most workstations in use today. As a result, integers were stored with only 2 bytes. It is not difficult to handle the conversion to longwords (4 byte integers) in `FORTRAN` . There is an unfortunate side effect to the use of the shorter integers: most of the real numbers (stored in 4 bytes as they are on 32-bit machines) do not begin on a longword (4-byte) boundary. While VAX systems will automatically compensate for this (thus providing backward compatibility to PDP-11's) through the `FORTRAN EQUIVALENCE` statement, most systems will not. While many vendors, such as SGI and SUN, provide a library of calls to handle misaligned data, the traps and handlers they use are very expensive at runtime. As a result, the read and write routines in GENAH simply use a `BYTE` array of length 512 for the header block. The parameters within the header are then individually equated with specific locations in the array, and the pass-by-reference standard of `FORTRAN-77` handles the values automatically. While this makes for lengthy coding in those subroutines, it works quickly and efficiently at runtime.

All of the parameters stored in the GENAH header are listed below in

Table B.1. This is followed by an in-depth discussion of each parameter and its use.

B.2 Description of Header Parameters

The following is a detailed description of each of the parameters currently defined in the GENAH file header. Each parameter is itemized by its FORTRAN name as it appears in the GENAH code. The parameter's function is described, followed by the range of allowable values, the number of bytes (in decimal) the variable is offset in the 512-byte header, and the FORTRAN variable type used to represent the parameter.

Parameters were listed strictly in offset order in the previous Table B.1 for reference purposes. Note that they are grouped here by *functionality*, not necessarily by order of their offset in the header. While the rough order of their appearance in the header is followed, the user will see deviations from that order when a parameter's function is more closely related to non-neighboring parameters.

B.2.1 Experiment, Target and File Description

These entries in the file header are used to provide an accurate description of the experimental run, the target under test, and the state of the data contained within the file. The descriptive strings can be whatever the user chooses; formation of an in-house standard for naming and numbering experiments and targets is recommended. The date and time entries refer to the file creation, not the execution of the experiment. They take the standard numerical forms `MM-DD-YY` and `HH:MM:SS`, although the date is actually numeric (integers), while the time is ASCII-encoded characters. This conforms to the `FORTRAN-77` system `TIME` and `DATE` utilities.

The radius of the target shell is included here since it forms a critical part of the description of the target.

The flag `NFLAG` is used at NRL to describe the status of the data. In combination with the file name extension, it indicates what processing has been completed on the data up to this file. It is purely an in-house standard; the user may use any code system desired. The codes used by NRL are described in Table B.2.1. One alternative suggestion would be to use the

Byte Offset	*Data Type*	*Name*	*Variable Description*
000	character*50	`TITLE`	Name, description of run
050	integer*2	`MONTH`	2-digit Month Code (01-12)
052	integer*2	`DAY`	2-digit Day Code (01-31)
054	integer*2	`YEAR`	2-digit Year Code (83-92)
056	character*2	`HOUR`	2-char. Hour Code (00-24)
058	character*2	`MINUTE`	2-char. MinuteCode (00-59)
060	character*2	`SECOND`	2-char. SecondCode (00-59)
062	character*38	`TARGET`	Name and description of target
100	integer*2	`NFQC`	*Center Frequency mantissa
102	integer*2	`NFQCXP`	*Center Frequency exponent
104	integer*2	`NUMREC`	# of Data Records in file
106	character*6	`DATATYPE`	6-character descriptor of data type
112	integer*2	`NDCVAL`	*# Bits DC Offset from calibration
114	integer*2	`NCKRT`	*Digitizer Rate mantissa
116	integer*2	`NEXP`	*Digitizer Rate exponent
118	integer*2	`NAVGS`	*# of records averaged per sample
120	integer*2	`NWVSCL`	*Peak Output Drive Voltage $\times 10$
122	integer*2	`IRATE`	*Drive Signal Output rate (us)
124	character*50	`SOURCE`	*Name of Drive Waveform file
174	integer*2	`NXO`	Sensor start position, x Coord. $\times 100$
176	integer*2	`NYO`	*Sensor start position, y Coord. $\times 100$
178	integer*2	`NZO`	*Sensor start position, z Coord. $\times 100$
180	integer*2	`NTO`	*Sensor start position, θ Coord. $\times 100$
182	integer*2	`NSCANS(3)`	# Scan surfaces in x, y, z or z, θ, r
188	integer*2	`NSTEPS(3)`	# Steps per scan in x, y, z or z, θ, r

Table B.1: NAH Data File Header Block Layout.
(*Note 1*: The header is a total of 512 bytes in length.)
(*Note 2*: A * indicates that the field is not always filled.)

Byte Offset	*Data Type*	*Name*	*Variable Description*
194	integer*2	IAXIS(3)	*Scan Axis Flags (Step, Scan, Surf.)
200	integer*2	IDIR(3)	*Start Direction Flags
206	real*4	DIST(3)	Step, Scan and Surface step sizes
218	character*18	STRING	*3 5-char. settling times between steps
236	character*20	MOVEFILE	*Name of Motion Control parameters file
256	integer*2	NBITS	*# of Bits in each Digitizer Channel
258	integer*2	NPOINTS	*# of Data points per Record
260	integer*2	NANGLE	*# of Angles in farfield recons.
264	(6 Empty Bytes)		
268	integer*2	NSCNTP	*Scan Type Code
270	real*4	GAINS(2)	*Total Channel Gain
278	real*4	SNSTVT(2)	*Transducer Sensitivity
286	real*4	XSTART3D	*Origin value of 1st Data Coordinate
290	real*4	DX3D	*Interval value of 1st Data Coordinate
294	real*4	YSTART3D	*Origin value of 2nd Data Coordinate
298	real*4	DY3D	*Interval value of 2nd Data Coordinate
302	real*4	ZSTART3D	*Origin value of 3rd Data Coordinate
306	real*4	DZ3D	*Interval value of 3rd Data Coordinate
310	real*4	FSTART	Frequency of first Bin
314	real*4	FDIF	Frequency interval between bins
318	real*4	RADIUS	Radius of target surface
322	real*4	DELTHETA	*Polar step angle in farfield recons.
326	real*4	DATAMAX	*Largest data value (mag.) in file
330	integer*2	IBINDATAMAX	*3rd Coord. Index of DATAMAX
332	integer*2	IXDATAMAX	*1st Coord. Index of DATAMAX
334	integer*2	IYDATAMAX	*2nd Coord. Index of DATAMAX
336	(172 Empty Bytes)		
508	integer*2	NBINS	Number of 3rd Coord. Intervals
510	integer*2	NFLAG	3-Digit Process Type Code

NAH Data File Header Block Layout, continued.

available 16 bits as individual flags indicating processing which has taken place so far on the data. For example, the first bit could indicate time or frequency domain, the second and third bits could indicate spatial or wavenumber domains, and so forth.

`TITLE` String containing the user-created name of an experiment run and possibly some description of it.
Allowable Values: Up to 50 alphanumeric (ASCII) characters
Byte offset: 000
Variable Type: Character*50
Criticality: Useful but not critical

`MONTH` 2-digit code representing the Month portion of the date and time the file was created.
Allowable Values: 01 through 12
Byte offset: 050
Variable Type: Integer*2
Criticality: Automatically generated by GENAH

`DAY` 2-digit code representing the Day portion of the date and time the file was created.
Allowable Values: 01 through 31
Byte offset: 052
Variable Type: Integer*2
Criticality: Automatically generated by GENAH

`YEAR` 2-digit code representing the Year portion of the date and time the file was created.
Allowable Values: 00 through 99
Byte offset: 054 *Variable Type:* Integer*2
Criticality: Automatically generated by GENAH

`HOUR` 2-character code representing the Hour portion of the date and time the file was created.
Allowable Values: ASCII characters 00 through 24
Byte offset: 056
Variable Type: Character*2
Criticality: Automatically generated by GENAH

MINUTE 2-character code representing the Minute portion of the date and time the file was created.
Allowable Values: ASCII characters 00 through 59
Byte offset: 058
Variable Type: Character*2
Criticality: Automatically generated by GENAH

SECOND 2-character code representing the Second portion of the date and time the file was created.
Allowable Values: ASCII characters 00 through 59
Byte offset: 060
Variable Type: Character*2
Criticality: Automatically generated by GENAH

TARGET String containing the user-created name of the target under study in current experiment and possibly a description of it.
Allowable Values: Up to 38 alphanumeric (ASCII) characters
Byte offset: 062
Variable Type: Character*38
Criticality: Useful but not critical

RADIUS Radius of cylindrical target's outer surface measured from the axis of symmetry in the current engineering units.
Allowable Values: Single-precision floating point value (positive)
Byte offset: 318
Variable Type: Real*4
Criticality: Critical

NFLAG 3-digit code representing the type of processing used to create data in this file. *Allowable Values:* (See Table B.2.1)
Byte offset: 510
Variable Type: Integer*2
Criticality: Automatically generated by GENAH

B.2.2 Number and Type of Data Records

These parameters describe the size of the file to the calling program. With knowledge of the number of records, the data type and the number of points

Flag Value	*Processing type*	*Data type*	*Processing Generation*
444	Structural Intensity	Complex*8	4
445	Structural Power	Complex*8	4
554	Acoustic Intensity $I(z, \phi; r')$	Complex*8	3
555	Velocity $V(z, \phi; r')$, $V(z, \phi'; r')$	Complex*8	3
556	Pressure $P(z, \phi; r')$, $P(k_z, n; r_0)$	Complex*8	3
557	Velocity $V(k_z, n; r')$	Complex*8	3
558	Velocity $V(z, n; r')$	Complex*8	3
559	Pressure $p(k_z, \phi; r_0)$	Complex*8	3
666	Pressure $P(z, \phi; r_0)$	Complex*8	2
776	Force Reference `.REF`	Complex*8	1
777	Calibrated Pressure `.CPX`	Complex*8	1
887	Raw A/D, Pressure `.PRS`	Integer*2	0
888	Raw A/D, 2 Channels `.SCN`	Integer*2	0
99x	Scattered Holograms	Complex*8	3

Table B.2: Values for processing type code `NFLAG`.

per record, the routine knows the size of the file. In fact, this is not often necessary for follow-on processing, so these parameters can be considered optional. The parameter `NPOINTS` is usually used to store the number of time samples per record in the original acquisition of hologram data. This can be used along with the sample rate to determine the frequency bin width and available frequency range.

`NUMREC` Total number of data records in file.
Allowable Values: 1 through 32767
Byte offset: 104
Variable Type: Integer*2
Criticality: Critical

`DATATYPE` 6-character descriptor of data word type.
Allowable Values: `CMPLX` for Complex*8 data, `REAL` for Real*4 data, `INTGR4` for Integer*4 or `INTGR2` for Integer*2
Byte offset: 106
Variable Type: Character*6
Criticality: Useful but not critical if data type is known

NPOINTS Number of digitized time samples in original data.
Allowable Values: 1 through 32767
Byte offset: 258
Variable Type: Integer*2
Criticality: Critical

B.2.3 Digital Acquisition Parameters

With the exception of NCKRT, these parameters are used only when the data are not yet calibrated. They provide all of the channel information necessary to convert from digital samples to engineering units. If the data have already been calibrated (usually the case), only NCKRT is needed. It gives the digitization period (or rate) in microseconds; half of the inverse of this number represents the highest allowable frequency (seldom the highest *useful* frequency) in a data set.

NDCVAL Number of bits offset for zero-volt input (DC) at A/D input, from calibration.
Allowable Values: 0 through 32767
Byte offset: 112
Variable Type: Integer*2
Criticality: Not critical if data are already calibrated

NCKRT Mantissa of input A/D digitizer clock rate in microseconds.
Allowable Values: 1 through 32767
Byte offset: 114
Variable Type: Integer*2
Criticality: Critical

NEXP Exponent of input A/D digitizer clock rate.
Allowable Values: -32767 through 32767, usually -6 through -3
Byte offset: 116
Variable Type: Integer*2
Criticality: No longer used; rate is assumed to be in microseconds

NAVGS Number of time series averaged per sample.
Allowable Values: 1 through 32767
Byte offset: 118

Variable Type: Integer*2
Criticality: Not critical

`NBITS` Number of bits in each digitizer channel.
Allowable Values: 10 through 16
Byte offset: 256
Variable Type: Integer*2
Criticality: Not critical if data are already calibrated

`GAINS(2)` Total channel gain from transducer to A/D, two channels.
Allowable Values: Single-precision floating point value
Byte offset: 270
Variable Type: Real*4
Criticality: Not critical if data are already calibrated

`SNSTVT(2)` Transducer sensitivity in engineering units, two channels.
Allowable Values: Single-precision floating point value
Byte offset: 278
Variable Type: Real*4
Criticality: Not critical if data are already calibrated

B.2.4 Source Waveform Information

These parameters characterize the source signal used to drive the target or to insonify it. None of these values is critical to GENAH since the source characteristics have already been convolved out of the pressure signal before reconstructions are begun. They are carried in the header mostly as a reference for the user.

When broadband nearfield acoustical holography was first developed, the drive waveform was a simple pulse, constructed to have a very broad frequency response. This response was the main lobe of a Kaiser-Bessel windowed sinusoid. As such, the lobe was well characterized by its center frequency. This value is contained in the two parameters `NFQC` and `NFQCXP`. (Note that the use of integers to store real values in mantissa-exponent form predates use of the VAX and addition of other, real-valued parameters to the header.) The center frequency of the drive waveform is computed as $f_c = \mathtt{NFQC} \times 10.0^{\mathtt{NFQCXP}}$. Note that recent holograms have been generated using chirped pulses with a nearly flat response across the band of interest. As a result, these parameters are seldom used anymore.

The output drive voltage is measured after the last stage of amplification, that is, it is the actual voltage appearing at the driver input. The integer value in **NWVSCL** must be divided by 10 to get the value in volts. The drive signal output rate refers to the clock rate, in microseconds, of the D/A converter used by the arbitrary waveform generator. Finally, if the arbitrary waveform is stored in a computer file, the name of that file can appear in the string **SOURCE**.

NFQC Mantissa of the source waveform center frequency.
Allowable Values: 0 through 32767
Byte offset: 100
Variable Type: Integer*2
Criticality: Not critical, no longer used

NFQCXP Exponent of the source waveform center frequency.
Allowable Values: -32767 through 32767 (usually less than 10)
Byte offset: 102
Variable Type: Integer*2
Criticality: Not critical, no longer used

NWVSCL Peak-to-peak output drive voltage $\times 10$.
Allowable Values: 0 through 32767
Byte offset: 120
Variable Type: Integer*2
Criticality: Not critical

IRATE Aribitrary waveform generator output clock rate (μs).
Allowable Values: 0 through 32767
Byte offset: 122
Variable Type: Integer*2
Criticality: Not critical

SOURCE Name of file containing Drive Waveform.
Allowable Values: Up to 50 alphanumeric (ASCII) characters
Byte offset: 124
Variable Type: Character*50
Criticality: Not critical

B.2.5 Hologram Aperture Information

As might be expected, a large number of parameters are used to describe the type, size and extent of the hologram aperture itself. Many of the parameters date back to a period when many different types of apertures were being explored, including planar, modified planar, cylindrical and multiple surface scans. To accomodate all of the options considered possible at that time, an extensive set of parameters was included in the header so that a "smart program" could reconstruct the aperture if necessary. Now that the bulk of experiments (particularly those analyzed using GENAH) are simple cylindrical single–surface scans, several of the parameters have fallen into disuse. This is noted where it applies.

Of all these parameters, the most important ones include `NXO`, which is used to represent the standoff distance (known as `DRHX`); `NSTEPS`, the number of points in each coordinate comprising the mesh; and `DIST`, the point separation distances in each coordinate of the mesh. Without these values, reconstruction cannot take place.

These parameters depend on the scans occurring in separable coordinate systems with equally-spaced points, as does GENAH itself. For conformal holography scans, where the mesh bends around the target and the step sizes change, the parameter `MOVEFILE` becomes critical, because the file named by this string is used to hold the point coordinate locations. Again, this is not as important for GENAH .

`NXO` Sensor start position in x or radial coordinate $\times 100$.
Allowable Values: -32767 through 32767 (positive when radial)
Byte offset: 174
Variable Type: Integer*2
Criticality: Critical: used to store radial standoff, `DRHX`

`NYO` Sensor start position in y or axial coordinate $\times 100$.
Allowable Values: -32767 through 32767
Byte offset: 176
Variable Type: Integer*2
Criticality: Not critical, seldom used

`NZO` Sensor start position in z coordinate $\times 100$.
Allowable Values: -32767 through 32767

Byte offset: 178
Variable Type: Integer*2
Criticality: Not critical, seldom used

`NTO` Sensor start position in θ coordinate $\times 100$.
Allowable Values: -32767 through 32767
Byte offset: 180
Variable Type: Integer*2
Criticality: Not critical, seldom used

`NSCANS(3)` Number of scan surfaces in x, y, z or z, θ, r.
Allowable Values: 1 through 32767, usually 1 or 2
Byte offset: 182
Variable Type: Integer*2
Criticality: Not critical, seldom used

`NSTEPS(3)` Number of steps per scan in x, y, z or z, θ, r.
Allowable Values: 1 through 32767
Byte offset: 188
Variable Type: Integer*2
Criticality: First and second values critical

`IAXIS(3)` Scan axis flags in (1) step direction, (2) scan line direction, (3) surface direction.
Allowable Values: $1 = x$, $2 = y$, $3 = z$, $4 = \theta$, $5 = r$
Byte offset: 194
Variable Type: Integer*2
Criticality: Not critical if type of scan known

`IDIR(3)` Start direction flags in (1) step direction, (2) scan line direction, (3) surface direction.
Allowable Values: 1 in positive coordinate direction, -1 in negative coordinate direction
Byte offset: 200
Variable Type: Integer*2
Criticality: Not critical if orientation of scan is known

`DIST(3)` Distance between mesh locations in (1) step direction, (2) scan line direction, (3) surface direction, in engineering units.

Allowable Values: Single-precision floating-point number
Byte offset: 206
Variable Type: Real*4
Criticality: First and second values critical

`STRING` Three 5-character strings representing the time allowed for settling between steps in (1) step direction, (2) scan line direction, (3) surface direction in the format `MM:SS`.
Allowable Values: `00:00` through `99:59` in ASCII numerics
Byte offset: 218
Variable Type: Character*18
Criticality: Not critical, seldom used

`MOVEFILE` Name of file containing robotic manipulator's motion control parameters.
Allowable Values: Up to 20 alphanumeric (ASCII) characters
Byte offset: 236
Variable Type: Character*20
Criticality: Not critical

`NSCNTP` Code indicating type of scan.
Allowable Values: 10 for single line scan, 20 for single surface (multiple lines) scan, 30 for multiple surface scan
Byte offset: 268
Variable Type: Integer*2
Criticality: Not critical, no longer used

B.2.6 Farfield Reconstruction Angle Information

These two parameters are not used by GENAH . They are reserved for post-processing in which holographic data is projected to a circle in the farfield. They are listed here for completeness.

`NANGLE` Number of angles computed.
Allowable Values: 1 through 32767
Byte offset: 260
Variable Type: Integer*2
Criticality: Farfield reconstructions only; not used in GENAH

DELTHETA Polar angle step size.

Allowable Values: Single-precision floating-point value

Byte offset: 322

Variable Type: Real*4

Criticality: Farfield reconstructions only; not used in GENAH

B.2.7 Data Block Descriptors

One can picture the data in a holography file as a large cube or block. Two of the coordinates of the cube represent the spatial dimensions spanned by the hologram aperture. The third coordinate represents the frequency range over which the holograms exist. Since all three coordinates are independent under Fourier Transform, each can be replaced by its transformed coordinate. For a program handling this "data block", it is important to have information describing the coordinate origin (the "corner" of the block) and the sizes of the discrete steps along each coordinate. The first six parameters of this group fall into that category. For historical reasons, the next two parameters provide that information redundantly in the most common case of holograms at individual frequencies. FSTART is the center frequency of the first hologram, and FDIF is the frequency interval between bins.

The next four parameters deal with the magnitude and location in the data block of the global maximum of the file. These parameters are not explicitly required by GENAH , nor are they set by it. However, they often prove useful for post-processing which might include visualization, for which scaling to the global maximum is helpful. If it is provided in the header, then the (somewhat expensive) search for the maximum need only be accomplished once regardless of how many times the data are later processed.

The last parameter, NBINS, is a critical one for GENAH . It contains the number of frequency bins (holograms) present in the data file. While it might be considered redundant with NPOINTS, with use at NRL the former has come to be associated with frequency bins while the latter is associated with the number of time samples per record taken in the original experiment. The key difference is that while NPOINTS is held to the same value throughout processing steps, NBINS reflects the current number of frequency bins (or time steps in the case of back-transforming to the time domain) actually contained in the data file. Since the range of bins can be widely different from one file to the next, NBINS becomes a critical means of determining the extent of the data in the current file.

`XSTART3D` Origin value of first data coordinate.
Allowable Values: Single-precision floating-point value
Byte offset: 286
Variable Type: Real*4
Criticality: Not critical, not used or set by GENAH

`DX3D` Interval value of first data coordinate.
Allowable Values: Single-precision floating-point value
Byte offset: 290
Variable Type: Real*4
Criticality: Not critical, not used or set by GENAH

`YSTART3D` Origin value of second data coordinate.
Allowable Values: Single-precision floating-point value
Byte offset: 294
Variable Type: Real*4
Criticality: Not critical, not used or set by GENAH

`DY3D` Interval value of second data coordinate.
Allowable Values: Single-precision floating-point value
Byte offset: 298
Variable Type: Real*4
Criticality: Not critical, not used or set by GENAH

`ZSTART3D` Origin value of third data coordinate.
Allowable Values: Single-precision floating-point value
Byte offset: 302
Variable Type: Real*4
Criticality: Not critical, not used or set by GENAH

`DZ3D` Interval value of third data coordinate.
Allowable Values: Single-precision floating-point value
Byte offset: 306
Variable Type: Real*4
Criticality: Not critical, not used or set by GENAH

`FSTART` Frequency of first hologram bin.
Allowable Values: Single-precision floating-point value
Byte offset: 310
Variable Type: Real*4
Criticality: Critical

`FDIF` Frequency interval between hologram bins.
Allowable Values: Single-precision floating-point value
Byte offset: 314
Variable Type: Real*4
Criticality: Critical

`DATAMAX` Maximum data value (magnitude) in file.
Allowable Values: Single-precision floating-point value
Byte offset: 326
Variable Type: real*4
Criticality: Not critical, not used or set by GENAH

`IBINDATAMAX` Third coordinate index of `DATAMAX`.
Allowable Values: 1 to `NBINS`
Byte offset: 330
Variable Type: Integer*2
Criticality: Not critical, not used or set by GENAH

`IXDATAMAX` First coordinate index of `DATAMAX`.
Allowable Values: 1 to `NSTEPS(1)`
Byte offset: 332
Variable Type: Integer*2
Criticality: Not critical, not used or set by GENAH

`IYDATAMAX` Second coordinate index of `DATAMAX`.
Allowable Values: 1 to `NSTEPS(2)`
Byte offset: 334
Variable Type: Integer*2
Criticality: Not critical, not used or set by GENAH

`NBINS` Number of holograms, time steps, or third coordinate intervals.
Allowable Values: 1 to `NPOINTS`
Byte offset: 508

Variable Type: Integer*2
Criticality: Critical

Appendix C

List of Inputs to GENAH

This appendix is a compilation of GENAH input parameters and allowed responses to them. Explanations of the parameters and their effects are not provided here. For more detail, a full description is available in Chapter 5.

Parameter:	`iconj`	Incoming data Conjugation flag
Allowable	`1`	Conjugate all incoming data ($e^{(i\omega t)}$)
Responses:	`0`	Do not conjugate incoming data ($e^{(-i\omega t)}$)

Paramter:	`ivax`	Binary number format flag
Allowable	`1`	Convert from DEC format to IEEE format.
Responses:	`0`	Do not perform any number format conversions.

Parameter:	`fname`	Input Filename
Allowable	(Unix)	`name` *or*
Responses:		`./name` *or*
		`dir/name` *or*
		`dir/subdir/.../name` *or*
		`/dir/subdir/.../name` *or*
		`/partition/dir/subdir/.../name` etc.
	(VMS)	`name.ext` *or*
		`[dir]name.ext` *or*
		`[dir.subdir...]name.ext` *or*
		`disk:[dir.subdir...]name.ext` *or*

		`machine::disk:[dir.subdir...]name.ext` etc. (String, $\leq$ 80 characters)
Parameter:	`units`	Type of Units for data in input file
Allowable	0	Inches (English)
Responses:	1	Meters (MKS)
	2	Centimeters (cgs)
Parameter:	`ncirpts`	Number of scan lines per hologram (Prompted only when header value is not an even number.)
Allowable Responses:	"n"	Actual number of scan lines in file (EVEN Integer, power of 2)
Parameter:	`window_choice`	Choice for Axial Data Window type
Allowable	0	No window (Rectangular)
Responses:	1	Tukey with 8-point taper
	2	Hanning (cosine squared)
	3	Hamming (cosine)
Parameter:	`izadd`	1st coordinate (axial) Zero Padding
Allowable	1	Yes, zero-pad input data
Responses:	0	No, do not zero-pad input data
Parameter:	`ihstart`	Starting hologram index (1st frequency bin)
Allowable Responses:	"ℓ"	Index of first frequency bin to be processed from the available set (positive integer)
Parameter:	`ihstep`	Hologram index step interval
Allowable Responses:	"m"	Increment to the next successive frequency bin to be processed from the available set (positive integer)

Parameter: `ihno`		Number of holograms to process
Allowable Responses:	"n"	Total number of frequency bins to be processed from the available set (positive integer)

Parameters: `rkc, alpha`		Wavenumber domain filter parameters
Allowable Responses:	"0, 0"	Do not apply any wavenumber filter
	"-1, 0"	Read in a frequency-dependent filter from an external file
	"a, b"	$k_{r(cutoff)}$ radians per length, Slope

Parameter: `wname`		Frequency-dependent wavenumber filter File Name
Allowable Responses:	"`name.ext`"	Full path and file name of file which contains parameters for filter set. (String, $\leq$ 80 characters)

Parameter: `nfilt`		Circumferential Order Filtering flag
Allowable Responses:	0	Include all available circumferential orders (No filtering)
	1	Remove circumferential orders above a set maximum

Parameter: `nf_max`		Maximum circumferential order cutoff (`nfilt` = 0 only)
Allowable Responses:	"n_{max}"	The maximum circumferential order included in reconstruction (positive integer, $\leq$ `ncirpts`/2

Parameter: `ifqout`		GENAH Output Processing Type
Allowable Responses:	0	Normal Reconstruction (P, V, I)
	1	K-space velocity *without* propagation (K)
	2	K-space velocity *with* propagation (G)
	3	$V(z, n)$ (NO inverse transform on ϕ) (N)
	4	$P(k_z, \phi)$ (*no* propagation, *no* transform on ϕ) (Q)
	5	$V(z, \phi')$ for orders $n = n_{min}, \ldots, n_{max}$.

Parameters: `nmin, nmax`		Minimum and Maximum Circumferential orders to keep in processing
Allowable Responses:	"n_{min}, n_{max}"	Lowest and highest orders to retain (positive integers; $0 \leq n_{min} < n_{max} \leq$ `ncirpts`/2

Parameter: `ichng`		Processing parameters change flag
Allowable Responses:	`0`	The reported parameters are correct: Proceed with reconstruction.
	`1`	The reported parameters need to be altered.

Parameter: `alat`		Axial Lattice Spacing (`ichng` = 0 only)
Allowable Responses:	"a"	Axial lattice spacing in units "`units`" (Positive real number)

Parameter: `drhx`		Standoff Distance (`ichng` = 0 only)
Allowable Responses:	"b"	Distance, in "`units`", from hologram surface to the desired reconstruction surface (Positive real number)

Parameter: `radius`		Reconstruction Radius (`ichng` = 0 only)
Allowable Responses:	"c"	The radius of the desired reconstruction surface in units "`units`" (Positive real number)

Parameter: `fname1`		Output File Name
Allowable Responses:	`name.*mn` `name.*vw`	$mn = 10 \times m.n = 10 \times k_{r(cutoff)}$ "vw" for *v*ariable *w*indow, frequency-dependent wavenumber filter) (Note that "*" is literal here) (String, $\leq$ 80 characters)

Appendix D

List of Routines in GENAH

This section contains a list of all the routines needed to run GENAH . Except for **genah** itself, which appears first, they are listed in alphabetical order. Each description contains a short definition, any important points to be noted, lists of arguments, input and output parameters and routines called. The latter gives the file dependencies and are listed in the order in which they appear in the calling routine. All routines are contained in separate files.

Note that only the master Bessel function routines `DBESJ`, `DBESY` and `DBESK` are listed; their dependent subroutines are not. They are all members of the `VAXMATH` package, where they are fully annotated. To repeat them here would waste space and detract from the intent of listing GENAH routines. Likewise, the many implicit `FORTRAN-77` functions called by GENAH are not listed.

program genah Routine for processing Cylindrical Nearfield Acoustical Holograms.

Arguments: None

Input Parameters: As described in Appendix C

Output Parameters: As described in Chapter 4

Routines Called: `filread_ieee, tukey, hanning, write_to_ihead, filwrite_ieee, wtrev, fft, windo, cang, DBESJ, DBESY, DBESK`

real function cang Function to compute the phase angle (in degrees) of a complex number

Note: This function is the analog of the `FORTRAN` built-in function `cabs`. An offset of 10^{-32} is added to each part to obviate the indeterminate case $tan^{-1}(0/0)$. Note that some systems do define this case for their C language implementations of `atan2`, but the `FORTRAN-77` standard does not.

Arguments: `z`

Input Parameters:

`z` (Complex*8) Complex number whose phase is to be computed

Output Parameters:

`cang` (Real*4) Phase angle (in degrees) of complex number `z`

Routines Called: None

subroutine cvt_2byte_to_int4 Routine to convert two consecutive bytes into an integer*4 value

Arguments: `int1, byte1, ieee`

Input Parameters:

`byte1` (Byte array, 2) Consecutive bytes to be converted

`ieee` (Integer*4) Number format conversion flag

- **1** Convert data from DEC to IEEE number format
- **0** Data is not converted (can be DEC or IEEE)

Output Parameters:

`int1` (Integer*4) Resulting integer value

Routines Called: None

subroutine cvt_4byte_to_real4 Routine to convert four consecutive bytes into a real*4 value

Arguments: `r1, byte1, ieee`

Input Parameters:

`byte1` (Byte array, 4) Consecutive bytes to be converted

`ieee` (Integer*4) Number format conversion flag

- **1** Convert data from DEC to IEEE number format

0 Data is not converted (can be DEC or IEEE)

Output Parameters:

`r1` (Real*4) Resulting single precision floating point value

Routines Called: None

subroutine cvt_decf_ieeef Routine to convert an array of DEC single precision floating point numbers to IEEE floating point.

Arguments: `dec_fp, ieee_fp, num`

Input Parameters:

`dec_fp` (Real*4, `num`) Array of DEC floating-point numbers

`num` (Integer*4) Number of values to be converted

Output Parameters:

`ieee_fp` (Real*4, `num`) Array of converted, word-swapped IEEE formatted numbers

Routines Called: None

subroutine cvt_int4_to_2byte Routine to convert an integer*4 value into two consecutive bytes

Arguments: `int1, byte1, ieee`

Input Parameters:

`int1` (Integer*4) Integer value to be converted

`ieee` (Integer*4) Number format conversion flag

1 Convert data from DEC to IEEE number format

0 Data is not converted (can be DEC or IEEE)

Output Parameters:

`byte1` (Byte array, 2) Consecutive bytes converted from integer

Routines Called: None

subroutine cvt_real4_to_4byte Routine to convert a real*4 value into four consecutive bytes

Arguments: `r1, byte1, ieee`

Input Parameters:

`r1` (Real*4) Single precision floating point value to be converted

`ieee` (Integer*4) Number format conversion flag

1 Convert data from DEC to IEEE number format

0 Data is not converted (can be DEC or IEEE)

Output Parameters:

`byte1` (Byte array, 4) Consecutive bytes converted from floating point value

Routines Called: None

subroutine DBESJ Routine to compute an n member sequence of the double precision Bessel Function of the First Kind, $J_\alpha(x)$ for non-negative α and double precision argument.

Note: VAXMATH routine

Arguments: `darg, alpha, maxn, djarray, nzero`

Input Parameters:

`darg` (Real*8) Double precision argument, $x \geq 0$

`alpha` (Real*8) Double precision order of first member in sequence, $\alpha \geq 0$

`maxn` (Integer*4) Number of members in sequence, $n \geq 1$

Output Parameters:

`djarray` (Real*8) Double precision vector result; the first `maxn` components contain values for $J_{\alpha+k-1}(x), k = 1, \ldots, \mathtt{maxn}$

`nzero` (Integer*4) Number of components of `djarray` set to zero due to underflow.

Routines Called: `DASYJY, DJAIRY, DLNGAM, D1MACH, I1MACH, XERROR`, all from VAXMATH

subroutine DBESY Routine to compute an n member sequence of the double precision Bessel Function of the Second Kind, $Y_\alpha(x)$ for non-negative α and double precision argument x.

Note: VAXMATH routine

Arguments: `darg, alpha, maxn, dyarray`

Input Parameters:

`darg` (Real*8) Double precision argument, $x \geq 0$

`alpha` (Real*8) Double precision order of first member in sequence, $\alpha \geq 0$

`maxn` (Integer*4) Number of members in sequence, $n \geq 1$

Output Parameters:

`dyarray` (Real*8) Double precision vector result; the first `maxn` components contain values for $Y_{\alpha+k-1}(x), k = 1, \ldots,$ `maxn`

Routines Called: `DASYJY, DBESY0, DBESY1, DBSYNU, DYAIRY, D1MACH, I1MACH, XERROR`, all from VAXMATH

subroutine DBESK Routine to compute an n member sequence of the double precision Modified Bessel Function of the Second kind, $K_\alpha(x)$ for non-negative α and double precision argument x.

Note: VAXMATH routine

Arguments: `darg, alpha, iflag, maxn, dkarray, nzero`

Input Parameters:

`darg` (Real*8) Double precision argument, $x \geq 0$

`alpha` (Real*8) Double precision order of first member in sequence, $\alpha \geq 0$

`iflag` (Integer*4) Scaling option flag; `iflag` $= 1$ returns $K_\alpha(x)$, `iflag` $= 2$ returns $e^x K_\alpha(x)$

`maxn` (Integer*4) Number of members in sequence, $n \geq 1$

Output Parameters:

`djarray` (Real*8) Double precision vector result; the first `maxn` components contain values for $K_{\alpha+k-1}(x), k = 1, \ldots,$ `maxn`

`nzero` (Integer*4) Number of components of `djarray` set to zero due to underflow when `iflag` $= 1$

Routines Called: `DASYIK, DBESK0, DBESK1, DBSKNU, DBSK0E, DBSK1E, D1MACH, I1MACH, XERROR`, all from VAXMATH

subroutine fft Routine to compute the Fast Fourier Transform of a complex vector

Note: Transform is computed *in place*

Arguments: `isw, n, l2n, ibr, i2, w, z`

Input Parameters:

`isw` (Integer*4) Forward/Inverse transform switch

0

1

`n` (Integer*4) Length of transform vector

`l2n` (Integer*4) log_2(n) (number of powers of 2)

`ibr` (Integer*4, `n`) Bit reversal index vector

`i2` (Integer*4, `l2n`) Power of 2 index vector

`w` (Complex*8, `n`/2) Exponential weighting factors

`z` (Complex*8, `n`) Vector to be transformed

Output Parameters:

`z` (Complex*8, `n`) Coefficients of transformed vector

Routines Called: None

subroutine filread_ieee Routine to open, read and close GENAH format data files.

Arguments: `iopen, lu, ieee, freq, filename, ihol, ldz, zc`

Input Parameters:

`iopen` (Integer*4) Routine control flag

1 Open the file and read header, put header information into the common block.

0 Read hologram number `ihol` and load data into `zc`

-1 Close the input file

`lu` (Integer*4) Logical unit for file to be opened/read

`ieee` (Integer*4) Number format conversion flag

1 Convert data from DEC to IEEE number format

0 Data is not converted (can be DEC or IEEE) `ldz` (Integer*4) Leading dimension of data array `zc`

`ihol` (Integer*4) Hologram number corresponding to current record(s)

Output parameters:

`freq` (Real*4) Frequency corresponding to `ihol`

`filename` (Character*80) Character string containing file name, passed back after `OPEN`

`zc` (Complex*8) Array of data values for bin `ihol` of file `filename`

Routines Called: `read_from_ihead, cvt_decf_ieeef`

subroutine filwrite_ieee Routine to open, write and close GENAH format data files.

Arguments: `iopen, lu, ihead, num_recs, filename, ihol, ldz, zc`

Input Parameters:

`iopen` (Integer*4) Routine control flag

- **1** Open the output file and write the header block.
- **0** Write hologram number `ihol` from array `zc`
- **-1** Close the output file

`lu` (Integer*4) Logical unit for file to be opened/written

`ihead` (Byte) 512–byte array with encoded header information

`num_recs` (Integer*4) Number of records to write, equivalent to number of scan lines (`ncirpts`)

`filename` (Character*80) Character string containing file name, passed back after `OPEN`

`ihol` (Integer*4) Hologram number corresponding to current record(s)

`ldz` (Integer*4) Leading dimension of data array `zc`

`zc` (Complex*8) Array of data values for bin `ihol` of file `filename`

Routines Called: None

subroutine hanning Routine to compute the coefficients for a Hanning window for an even sequence

Note: The Hanning window is one half-cycle of $cos^{\mathbf{order}}$

Arguments: `window, npts, order`

Input Parameters:

- `npts` (Integer*4) Length of window vector
- `order` (Integer*4) Power to which Cosine is raised

Output Parameters:

- `window` (Real*4, `npts`) Coefficients of Hanning window of length `npts`

Routines Called: None

subroutine read_from_ihead Routine to read the currently open file's header parameters from the byte array `ihead` into the header variables

Note: Call this routine *after* `ihead` is loaded from record 1 of a broadband holography data file.

Arguments: `ihead, ieee`

Input Parameters:

- `ieee` (Integer*4) Number format conversion flag
  - **1** Convert data from DEC to IEEE number format
  - **0** Data is not converted (can be DEC or IEEE)
- `ihead` (Byte array, 512) Raw bytes from record 1 of data file, declared in `include` file `header_ieee.f`

Output Parameters:

- `header_ieee_char` Common block containing all header parameters which are character strings, declared in `include` file `header_ieee.f`
- `header_ieee` Common block containing all header parameters which are not character strings, declared in `include` file `header_ieee.f`

Routines Called: `cvt_2byte_to_int4, cvt_4byte_to_real4`

subroutine tukey Routine to compute the coefficients for a flat-top Tukey window

Note: This version is hardwired to produce an 8 point taper.

Arguments: `window, npts`

Input Parameters:

`npts` (Integer*4) Length of window vector

Output Parameters:

`window` (Real*4, `npts`) Coefficients of 8 point taper Tukey window of length `npts`

Routines Called: None

real function windo Routine to compute the modulus of the two dimensional wavenumber domain filter

Arguments: `rkc, efold, xk, yk`

Input Parameters:

`rkc` (Real*4) Cutoff wavenumber (radial)

`efold` (Real*4) Exponential folding rate (steepness of the filter)

`xk` (Real*4) First coordinate wavenumber (k_x or k_z)

`yk` (Real*4) Second coordinate wavenumber (k_y or n/a)

Output Parameters:

`windo` (Real*4) Modulus of filter at $k_r = \sqrt{k_x^2 + k_y^2}$

subroutine write_to_ihead Routine to write the current header parameters from the header variables into the byte array `ihead`

Note: Call this routine *before* `ihead` is written to record 1 of a broadband holography data file.

Arguments: `ihead, ieee`

Input Parameters:

`ieee` (Integer*4) Number format conversion flag

1 Convert data from DEC to IEEE number format

0 Data is not converted (can be DEC or IEEE)

`header_ieee_char` Common block containing all header parameters which are character strings, declared in `include` file `header_ieee.f` header_ieee Common block containing all header parameters which are not character strings, declared in include file header_ieee.f

Output Parameters:

ihead (Byte array, 512) Raw bytes for record 1 of data file, declared in include file header_ieee.f

Routines Called: cvt_int4_to_2byte, cvt_real4_to_4byte

subroutine wtrev Routine to compute Fast Fourier Transform weighting factors and bit reversal index

Arguments: isw, n, l2n, ibr, i2, w

Input Parameters:

isw (Integer*4) Forward/Inverse transform switch

0

1

n (Integer*4) Length of transform vector

l2n (Integer*4) log_2(n) (number of powers of 2)

Output Parameters:

ibr (Integer*4, n) Bit reversal index vector

i2 (Integer*4, l2n) Power of 2 index vector

w (Complex*8, n/2) Exponential weighting factors

Routines Called: None

Appendix E

Wavenumber Filters in GENAH

This section is intended to provide a concise description of the matched exponential filter used by GENAH .

Two parameters control the filter. The first, **rkc**, is the cutoff wavenumber, $k_{r(cutoff)}$. This is the radius of the circle on the two dimensional wavenumber domain where the filter reaches a value of 1/2 (or the "3 dB down point", as filter designers would call it). The "radial" (actually a trace) wavenumber for any frequency is defined as follows:

$$
\begin{aligned}
\Delta k_z &= \frac{2\pi}{N_z * a_z} \\
k_z &= i_z \Delta k_z \; ; \quad -N_z/2 \leq i_z < N_z/2 \\
k_\phi &= \frac{n}{r_0} \; ; \quad -n_{max}/2 \leq n < n_{max}/2 \\
k_r &= \sqrt{k_z^2 + k_\phi^2}
\end{aligned}
$$

where N_z is the number of points in the axial direction, a_z is the lattice spacing in the axial direction, n is the circumferential order, n_{max} is the maximum circumferential order and r_0 is the radius of the reconstruction surface. The filter magnitude for a given radial wavenumber is then defined as

$$
f(k_r) = 1 - \frac{e^{-x}}{2} \; ; \quad k_r < k_{r(cutoff)}
$$

$$= \frac{e^x}{2} \; ; \quad k_r \geq k_{r(cutoff)}$$

where

$$x = \frac{1}{\alpha}\,(1 - \frac{k_r}{k_{r(cutoff)}}) \; .$$

The second parameter controls the rate at which the filter transitions from 1 to 0 as a function of radial wavenumber. It is known as the steepness parameter or the exponential folding rate. As with any filter response, the steeper the roll-off, the higher the sidelobes or ringing in the inverse transform domain. Figure E.1 shows filters for a range of steepness parameters, α. Typically, $\alpha = 0.05$ is a good choice for most reconstructions.

When trying to determine values for the cutoff wavenumber, use of Processing Options 1 or 2 can be valuable (see Chapter 4). Plots of the wavenumber domain response will reveal a relatively circular region of low response. Inside this region lies the data from the shell itself. Outside this region lies high wavenumber noise, amplified by the propagator. Placing the cutoff wavenumber at the radius of this circular region is most effective. Since the valid data region often grows radially with increasing frequency, it is necessary to use the "Variable Window" option, specifying new filters as the frequency increases.

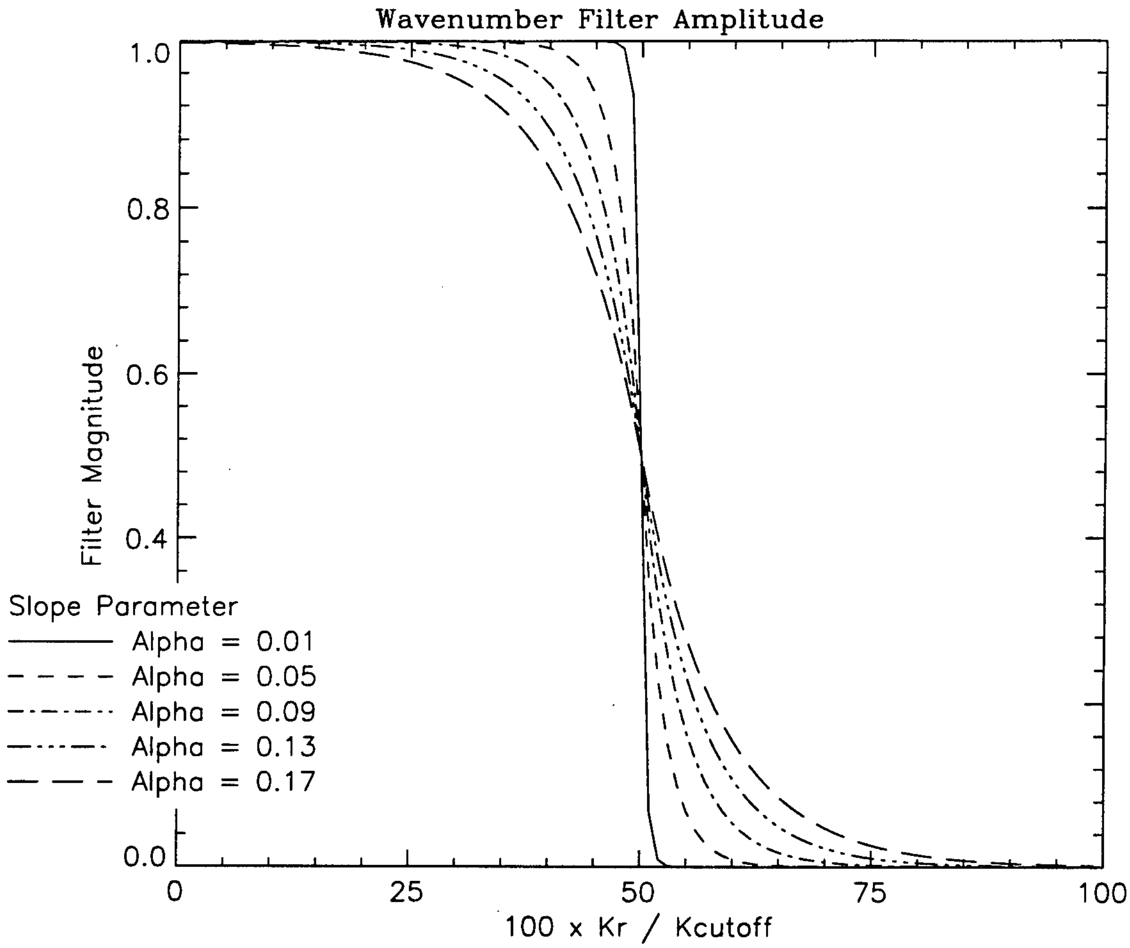

Figure E.1: Plot of a set of wavenumber domain filter responses for various values of the steepness parameter, α.

Bibliography

[1] E. G. Williams and J. D. Maynard, *Holographic imaging without the wavelength resolution limit*. Phys. Rev. Lett., **45**(7) 1980, p. 554.

[2] J. D. Maynard, E. G. Williams, and Y. Lee, *Nearfield acoustic holography: I. Theory of generalized holography and the development of NAH.* J. Acous. Soc. Amer. **78**(4) 1985, p. 1395.

[3] E. G. Williams, H. D. Dardy and K. B. Washburn, *Generalized nearfield acoustical holography for cylindrical geometry: theory and experiment.* J. Acous. Soc. Amer. **81**(2) 1987, p. 389.

[4] W. A. Veronesi and J. D. Maynard, *Nearfield acoustic holography: II. Holographic reconstruction algorithms and computer implementation.* J. Acous. Soc. Amer. **81**(5) 1987, p. 1307.

[5] P. M. Morse and H. Feshbach, **Methods of Theoretical Physics**. New York: McGraw-Hill, 1953, Volume 1, pp. 803-833.